Distributed Low-power Image Processing in Wireless Sensor Networks for Intelligent Video Surveillance Applications

by

Junbin Liu, BEng (Hons, 1st Class)

PhD Thesis

Submitted in Fulfilment

of the Requirements

for the Degree of

Doctor of Philosophy

at the

Queensland University of Technology

Image and Video Research Laboratory

Science and Engineering Faculty

June 2012

Keywords

Wireless Smart Camera, Distributed System, Embedded System, Pervasive Computing, Image Processing, Computer Vision, Projective Geometry, Camera Calibration, Object Localisation, Object Tracking, Camera Placement.

Abstract

Distributed Wireless Smart Camera (DWSC) network is a special type of Wireless Sensor Network (WSN) that processes captured images in a distributed manner. While image processing on DWSCs sees a great potential for growth, with its applications possessing a vast practical application domain such as security surveillance and health care, it suffers from tremendous constraints. In addition to the limitations of conventional WSNs, image processing on DWSCs requires more computational power, bandwidth and energy that presents significant challenges for large scale deployments. This dissertation has developed a number of algorithms that are highly scalable, portable, energy efficient and performance efficient, with considerations of practical constraints imposed by the hardware and the nature of WSN. More specifically, these algorithms tackle the problems of *multi-object tracking and localisation in distributed wireless smart camera networks* and *optimal camera configuration determination*.

Addressing the first problem of multi-object tracking and localisation requires solving a large array of sub-problems. The sub-problems that are discussed in this dissertation are *calibration of internal parameters*, *multi-camera calibration for localisation* and *object handover for tracking*. These topics have been covered extensively in computer vision literatures, however new algorithms must be invented to accommodate the various constraints introduced and required by the DWSC platform.

A technique has been developed for the automatic calibration of low-cost cameras which are assumed to be restricted in their freedom of movement to either pan or tilt movements. Camera internal parameters, including focal length, principal point, lens distortion parameter and the angle and axis of rotation, can be recovered from a minimum set of two images of the camera, provided that the axis of rotation between the two images goes through the camera's optical centre and is parallel to either the vertical (panning) or horizontal (tilting) axis of the image.

For object localisation, a novel approach has been developed for the calibration of a network of non-overlapping DWSCs in terms of their ground plane homographies, which can then be used for localising objects. In the proposed approach, a robot travels through the camera network while updating its position in a global coordinate frame, which it broadcasts to the cameras. The cameras use this, along with the image plane location of the robot, to compute a mapping from their image planes to the global coordinate frame. This is combined with an occupancy map generated by the robot during the mapping process to localised objects moving within the network.

In addition, to deal with the problem of object handover between DWSCs of non-overlapping fields of view, a highly-scalable, distributed protocol has been designed. Cameras that follow the proposed protocol transmit object descriptions to a selected set of neighbours that are determined using a predictive forwarding strategy. The received descriptions are then matched at the subsequent camera on the object's path using a probability maximisation process with locally generated descriptions.

The second problem of camera placement emerges naturally when these pervasive devices are put into real use. The locations, orientations, lens types etc. of the cameras must be chosen in a way that the utility of the network is maximised (e.g. maximum coverage) while user requirements are met. To deal with this, a

statistical formulation of the problem of determining optimal camera configurations has been introduced and a Trans-Dimensional Simulated Annealing (TDSA) algorithm has been proposed to effectively solve the problem.

Contents

List of Tables

List of Figures

List of Algorithms

Notation

Lower case letters, e.g. c	scalar value.
Lower case bold letters, e.g. $\mathbf{x}$	A column vector.
Upper case bold letters, e.g. $\mathbf{M}$	A matrix or a set.
Upper case letters, e.g. C	An element from a set.
$\mathbf{M}_{m \times n}$	A matrix with m rows and n columns.
$\mathbf{I}$	The identity matrix – a square matrix with the diagonal entries equal to one and all other entries equal to zero.
$\lvert\mathbf{M}\rvert$	The determinant of the matrix $\mathbf{M}$.
$\mathbf{0}$	The matrix whose elements are all zeros.

Acronyms & Abbreviations

2D	Two-dimensional
3D	Three-dimensional
6LowPan	Internet Protocol Version 6 over Low-power Wireless Personal Area Networks
BIP	Binary Integer Programming
CMU	Carnegie Mellon University
CSIRO	Commonwealth Scientific and industrial Research Organisation of Australia
CTP	Collection Tree Protocol
DWSC	Distributed Wireless Smart Camera
DSP	Digital Signal Processor
IAC	Image of the Absolute Conic
FoV	Field of View
FPS	Frames per Second
GMM	Gaussian Mixture Model
HMM	Hidden Markov Model
IVS	Intelligent Video Surveillance

MAP	Maximum A-Posteriori
MCMC	Markov Chain Monte Carlo
MCU	Memory Controller Unit
MSER	Maximally Stable Extremal Regions
MoG	Mixture of Gaussian
PTZ	Pan Tilt Zoom
QEP	Quadric Eigenvalue Problem
RANSAC	Random Sample Consensus
SA	Simulated Annealing
SIFT	Scale Invariant Feature Transform
SLAM	Simultaneous Localisation and Mapping
SPI	Serial Peripheral Interface
SURF	Speeded Up Robust Feature
SVD	Singular Value Decomposition
SVM	Support Vector Machine
TDSA	Trans-Dimensional Simulated Annealing
UCBerkley	University of California at Berkley
UCLA	University of California at Los Angeles
WMSN	Wireless Multimedia Sensor Network
WSC	Wireless Smart Camera
WSN	Wireless Sensor Network

List of Publications

Published at International Conferences and Workshops

P. Corke, **J. Liu**, D. Moore and T. Wark, "Design and evaluation of an image analysis platform for low-power, low-bandwidth camera networks," in *Proc. ImageSense'08: Workshop on Applications, Systems, and Algorithms for Image Sensing*, Raleigh, North Carolina, 2008.

J. Liu, D. O'Rourke, T. Wark, R. Lakemond and S. Sridharan, "Camera calibration in wireless multimedia sensor networks," in *Proc. Intelligent Sensors, Sensor Networks and Information Processing, International Conference on*, Melbourne, Australia, 2009, pp. 301–306. (Acceptance rate: 39%. **Best student paper award**.)

D. O'Rourke, R. Jurdak, **J. Liu**, D. Moore and T. Wark, "On the feasibility of using servo-mechanisms in wireless multimedia sensor network deployments," in *Local Computer Networks (LCN), IEEE 34th Conference on*, Zurich, Switzerland, 2009, pp. 826–833.

J. Liu, D. O'Rourke, T. Wark, S. Denman and S. Sridharan, "A distributed protocol for object tracking in wireless multimedia sensor networks," in *Proc. Intelligent Sensors, Sensor Networks and Information Processing, International*

Conference on, Brisbane, Australia, 2010, pp. 67–72.

D. O'Rourke, **J. Liu**, T. Wark, W. Hu, D. Moore, L. Overs and R. Jurdak, "Demo Abstract: Towards a framework for a versatile wireless multimedia sensor network platform," in *Proc. 9th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*, Stockholm, Sweden, 2010, pp. 412-413.

J. Liu, T. Wark, S. Martin, P. Corke and M. D'Souza, "Distributed object tracking with robot and disjoint camera networks," in *Proc. PerCom - WORKSHOPS: IEEE International Conference on Pervasive Computing and Communications Workshops*, Seattle, WA, 2011, pp. 380-383. (Acceptance rate for full and concise papers: 17.5%.)

Y. Liu, Q. Wang, **J. Liu** and T. Wark, "MCMC-based indoor localization with a smart phone and sparse WiFi access points," in *Proc. PerCom - WORKSHOPS: International Workshop on the Impact of Human Mobility in Pervasive Systems and Applications (PerMoby)*, Lugano, Switzerland, 2012.

Y. Shen, W. Hu, M. Yang, **J. Liu** and C. T. Chou, "Poster Abstract: Efficient background subtraction for tracking in embedded camera networks," in *Proc. Information Processing in Sensor Networks, IPSN'12*, Beijing, China, 2012.

Submitted to International Conferences and Workshops

J. Liu, C. Fookes, T. Wark and S. Sridharan, "On the statistical determination of optimal camera configurations in large scale surveillance networks," submitted to *European Conference on Computer Vision*, Firenze, Italy, 2012.

Y. Shen, W. Hu, **J. Liu**, M. Yang, B. Wei and C. T. Chou, "Efficient background subtraction for real-time tracking in embedded camera networks," submit-

ted to *10th ACM Conference on Embedded Networked Sensor Systems, Sensys'12*, Toronto, Canada, 2012

International Journals

J. Liu, T. Wark, R. Lakemond and S. Sridharan, "Self-Calibration of Wireless Cameras with Restricted Degrees of Freedom," *Computer Vision and Image Understanding*. (Accepted. Journal impact factor 2.53.)

Y. Liu, Q. Wang, **J. Liu**, T. Wark and J. Chen, "An automated and simple localization method for networked top-view disjoint cameras," submitted to *IEEE Transactions on Instrumentation & Measurement* in Feb 2012. (Impact factor 1.11.)

D. O'Rourke, **J. Liu**, N. Kottege, R. Jurdak, "A Versatile and Realistic Simulator for Wireless Multimedia Sensor Networks," submitted to *ACM Transactions on Sensor Networks* in May 2012. (Impact factor 2.28.)

Certification of Thesis

The work contained in this thesis has not been previously submitted for a degree or diploma at any other higher educational institution. To the best of my knowledge and belief, the thesis contains no material previously published or written by another person except where due reference is made.

Signed: QUT Verified Signature

Date: 13/02/2013

Acknowledgments

I would like to thank my supervisors, Prof. Sridha Sridharan, Dr. Tim Wark and Dr. Clinton Fookes, for their support and guidance throughout my PhD candidature. They provided access to excellent facilities and a knowledgeable and helpful group of colleagues.

I would like to thank my family for their support throughout my education. They have made it possible for me to pursue a career path that is well suited to my interests, ambitions and aptitude.

Finally, I would like to thank my colleagues in the SAIVT group (QUT) as well the Autonomous Systems Lab (CSIRO ICT Centre) and express my gratitude for their cooperative culture. Each one contributed to a productive and enjoyable work environment and provided support, insights, ideas and motivation on the path to completing this project.

Junbin Liu

Queensland University of Technology
June 2012

Chapter 1

Introduction

1.1 Overview

Recent technology advances in both short range radio communications and increasing semiconductor speeds have given rise to strong interest in wireless sensor networks (WSNs). Low-power, inexpensive, wireless capable sensors are embedded in the physical environment of interest, operating in a collaborative fashion to collect information overtime across a volume of space large enough to exhibit significant variations [26]. Typical applications of WSN include monitoring environment variables, such as temperature, soil moisture, or more advanced tasks including ecological habitat monitoring and environmental contamination detection [70, 101].

Whilst most WSN applications have focused on returning measured data values over networks, the increasing availability of low-cost, fast, digital signal processors has meant that sophisticated processing, typically image processing, is becoming a more feasible option for applications which were unlikely to have been con-

sidered previously. This gives rise to the Distributed Wireless Smart Camera (DWSC) networks [99], which have recently been the focus of research in a wide variety of areas including digital signal processing, communication, networking, control and statistics [18]. Distributed wireless smart cameras are vision systems capable of high-level interpretation of the environment, and they have the ability to communicate with one another or base stations via wireless links. The realisation of DWSC enables a wide variety of application areas as these smart devices combine video sensing, processing, and wireless communication on a single embedded platform. A large-scale distributed wireless smart camera network can cover a large building (such as an airport) and provide intelligent video surveillance of the area. On the other hand, smaller deployments can cover places such as homes or aged care facilities, which enable smart environment applications. Typical wireless smart cameras can be found in [20, 30, 84, 102, 123].

This dissertation is an investigation into how a number of important topics in video based surveillance can be translated into the domain of distributed wireless smart camera networks which have their unique challenges. These topics include camera calibration, distributed object localisation and tracking and optimal placement of cameras.

Camera calibration refers to the computation of the parameters in a camera model that describes its operation. More specifically these parameters are internal ones (focal length, principal point, skew, distortion, aspect ratio), which are invariant to deployments and external ones (position and orientation), which are deployment-specific parameters. Without these parameters, information in a camera view can hardly be transferred to another camera view or to a global coordinate frame. This is particularly important for object localisation and tracking. Traditionally, cameras can be calibrated to high accuracies but these methods, however, can not be translated into the DWSC platform simply due to the large

scale and frequently changing nature of the network. The prospect of automating the calibration process (for both internal and external parameters) is thus attractive and has inspired much research.

After the external and internal parameters of all the cameras in a DWSC network have been found, the network of cameras can be put into real use: object localisation and tracking. Object localisation refers to the finding of the real world locations of an object observed in a camera network while tracking refers to the association of observations of the same object across multiple camera views. With known camera parameters (or a reduced version: homography between the camera's image plane and the ground plane) object locations can be effectively computed using triangulation (Section 3.2.3) with two or more camera views or using homography (Section 3.2.3) with a single view. Tracking across multiple cameras, however, remains a challenging task for DWSC networks as overlapping camera views can not be assumed and therefore alternative ways of establishing object correspondences across different cameras such as path dependency, transition time measurements must be sought to assist the problem of tracking.

Deploying a large camera network in the first place is a difficult task and it is therefore highly preferable that the deployed network achieves optimal performance without the need for significant modifications. However, the current empirical approach to camera placement can not guarantee any form of optimality in terms of least number of cameras, maximum coverage etc. Thus, to fully utilise the potential of these large scale networks, an automated approach to finding the optimal camera parameters given a set of user requirements is in great demand. This problem is tackled as part of this thesis.

1.1.1 Motivation

Networks of cameras have been widely used in the area of intelligent video surveillance (IVS), especially in situations where the sensing area exceeds the capability of a single camera, or different views of the same scene are to be acquired. Based on user defined policies, IVS systems can automatically identify potential risks by detecting, localising, tracking and recognising targets and/or events of interest. Most of the current IVS systems follow a centralised architecture where video feeds from multiple cameras linked by high speed cable connections are processed in a server with significant computing power. These systems suffer from a number of draw-backs, such as high bandwidth and processing power requirement, which limits their scalability.

DWSC networks possess vast potentials. As a result of local processing and wireless communication, networks of wireless smart cameras are becoming increasingly popular due to improved scalability and flexibility, i.e. they can be quickly deployed in large numbers and redeployed to different locations in response to changes in user demands or the environment. While current technology in wired intelligent surveillance systems can achieve a satisfactory level of performance in a number of areas, they suffer from many deployment constraints, e.g. cable requirement for power supply and communication between sensors and fusion centres. DWSCs eliminate these problems by the use of reusable energy sources and wireless radio communications at the expenses of processing power. Another advantage of DWSCs for surveillance applications is that a large number of cameras can be deployed to provide greater coverage of the monitored space.

Distributed wireless smart camera networks in surveillance should be seen as an extension to current centralised surveillance technology since they allow certain tasks that were previously limited by the cable and/or power constraints to

be now feasible. For multi-camera applications, image processing migrates from centralised servers onto these pervasive sensors. For these platforms, it is very expensive to constantly stream video over a low-power wireless link to a centralised base station where fusion of information from multiple nodes normally occurs. Instead, the available low-power on-node processors must be able to process the video stream and extract high-level information and make it available to the end users or other nodes if the task requires collaboration among multiple cameras. Computer vision tasks such as motion segmentation, model based tracking have been studied extensively in the past decades for centralised architectures. However embedded image processing is still a relatively new field of research which has gained a lot of attention over the past few years as demonstrated by an increasing number of dedicated publication avenues. [1]

Despite the vast potential, algorithm design for this platform is severely constrained by the available hardware and distributed nature of the platform. For example, Due to the large number of cameras that exist in this type of networks, any algorithm must be autonomous to ensure the scalability and flexibility. Also algorithms must not rely on the global knowledge of the network; any camera can, at most, know the status of their nearby neighbours as any communication incurs some cost (energy, resources) that can not be underestimated (discussed in Section 2.3). These challenges necessitate the exploration of algorithms that fully exploit the potential of the platform, which motivates this dissertation.

[1] Example publication avenues include the International Conference on Distributed Smart Cameras (ICDSC) and IEEE Journal of Selected Topics in Signal Processing — Special Issue on Distributed Processing in Vision Networks.

1.2 Aims and Objectives

1.2.1 Scope

While image processing in WSNs sees a great potential for growth, with its applications possessing a vast practical application domain, it suffers from tremendous constraints. In addition to the limitations of conventional WSNs, DWSCs require more computational power, bandwidth and energy that presents significant challenges for large scale deployments. The main objective of this PhD project is to bridge this gap by developing WSN suitable image processing algorithms that are highly scalable, portable, efficient in energy and performance, with considerations of practical constraints imposed by the hardware and the nature of wireless sensor network. More specifically, these algorithms will tackle the problems of *multi-object tracking and localisation in distributed smart camera networks* and *optimal camera configuration determination.*

Addressing the first problem of multi-object tracking and localisation requires solving a large array of sub-problems, more specifically, the topics that are included in this dissertation are calibration of internal parameters, calibration of external parameters for localisation and object handover for tracking. These topics have been covered extensively in computer vision literatures, however new algorithms must be invented to accommodate the various constraints introduced and required by the distributed wireless smart camera platform.

The second problem of camera placement emerges naturally when these pervasive devices are put into real use. The location, the orientation, the lens type etc. must be chosen in a way that some kind of user requirement is optimised. Currently solutions to the camera placement problem rely on expert knowledge which can not guarantee the maximisation of the network's potential. On the

other hand, most automatic approaches are either not applicable to large scale camera networks or produce results that are far below optimal solutions. Thus, another focus of this dissertation is the proposal of a generalised approach to camera placement which is accurate, flexible and can deal with relatively large scale problems.

1.2.2 Objectives

The objectives of this dissertation are four-fold:

1. Self-calibration of the internal parameters of wireless smart cameras.

   Design an algorithm with can automatically compute the internal parameters of the cameras, without the need for manual input or specific calibration object. This objective has been achieved and is discussed in Chapter 4.

2. Multi-camera network calibration.

   Design an algorithm that is able to automatically find the relationship between cameras as well as each camera and the real-world scene. The method should be accurate and autonomous. This objective has been achieved and is discussed in Chapter 5.

3. Multi-camera tracking of objects.

   Design a scalable approach to deal with the problem of target handover between non-overlapping wireless smart cameras in a multi-object tracking scenario. This objective has been achieved and is discussed in Chapter 6.

4. Determination of optimal multi-camera configurations.

   Design an approach which can achieve the following task: given a user objective (or multiple objectives), find the optimal camera configurations that

satisfy a set of constraints. A example of this task can be the computation of the cameras' positions and orientations to achieve a 100% coverage while minimising the number of cameras used. While application specific, optimality in this particular case refers to the minimal number of cameras needed. The approach should be able to accommodate a large number of cameras, typically tens to hundreds but still offers adequate performance in terms of the optimality of the solution. This objective has been achieved and is further discussed in Chapter 7.

The linkages between the various objectives of this dissertation are summarised in Figure 1.1.

1.3 Organisation of the thesis

1.3.1 Chapter 2: Background on Distributed Wireless Smart Camera Platforms

This chapter provides a detailed introduction of existing wireless smart cameras (WSC), including the one used in this PhD research as well as a number of other ones designed by various research institutions over the past few years. All the platforms are compared according to a number of criteria, including processing speed, memory limit, communication bandwidth and energy cost of the platform. Following the description of the hardware platform is an introduction of a number of relevant concepts and unique features of the DWSC platform. Some of these concepts are critical to the understanding of the design methodologies and motivations behind the algorithms proposed in this thesis. These concepts include: difficulty in local execution, real-time requirement, multi-purpose, collaborative

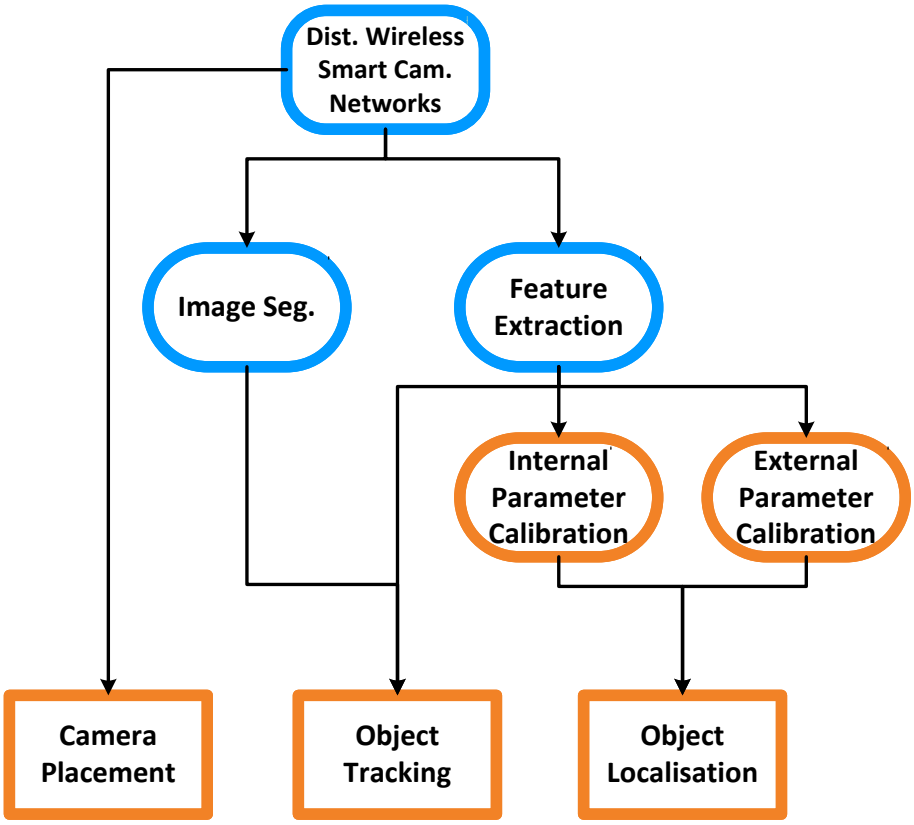


Figure 1.1: Linkage between the various objectives of this dissertation. The blocks with blue borders are fields of study involved in this dissertation and orange borders represent areas where contributions are made in this dissertation.

signal processing, autonomy and adaptation and user interaction.

1.3.2 Chapter 3: Background on Relevant Computer Vision Principles and Techniques

This chapter introduces some of the basic concepts in projective geometry and object tracking that are used throughout this dissertation. Projective geometry encompasses camera geometry and the formation of images through 3D to 2D projections. To obtain meaningful measurements from cameras, for example to

use a camera network for localisation of objects, the image formation process must be understood and the camera internal and external parameters must be sought. All these concepts are included in the first part of this chapter. In the second part of this chapter, a number of concepts that are extensively used in object detection, tracking and localisation are discussed. The subject is treated in a manner that provides the understanding and algebraic tools required to formulate and solve the problem of *camera internal parameter calibration, multi-camera calibration and tracking of objects*. The concepts discussed in this chapter include: homogeneous coordinates, projective transformations, pinhole camera model and camera calibration, foreground segmentation, object detection, camera geometry in object tracking and localisation and multi-camera tracking architectures.

1.3.3 Chapter 4: Self-calibration of the Internal Parameters of Wireless Smart Cameras

This chapter presents an approach for the automatic calibration of low-cost cameras which are assumed to be restricted in their freedom of movement to either pan or tilt movements. Camera parameters, including focal length, principal point, lens distortion parameter and the angle and axis of rotation, can be recovered from a minimum set of two images of the camera, provided that the axis of rotation between the two images goes through the camera's optical centre and is parallel to either the vertical (panning) or horizontal (tilting) axis of the image. Previous methods for auto-calibration of cameras based on pure rotations fail to work in these two degenerate cases. In addition, the proposed approach includes a modified RANdom SAmple Consensus (RANSAC) algorithm, as well as improved integration of the radial distortion coefficient in the computation of inter-image homographies. It is shown that these modifications are able to increase the overall efficiency, reliability and accuracy of the homography compu-

tation and calibration procedure using both synthetic and real image sequences.

1.3.4 Chapter 5: Autonomous Calibration of Wireless Smart Camera Networks and Object Localisation

Chapter 5 presents a novel approach for calibration of a network of distributed wireless smart cameras covering a large observation area with non-overlapping fields of view, which can then be used for localising and tracking objects. With the aid of a low-cost mobile robot, an innovative approach is used for camera calibration and object tracking. Most current methods to camera network calibration involve human input, limiting the scalability and flexibility of the networks. In the proposed approach, a robot travels through the camera network while updating its position in a global coordinate frame, which it broadcasts to the cameras. The cameras use this, along with the image plane location of the robot, to compute a mapping from their image planes to the global coordinate frame. This is combined with an occupancy map generated by the robot during the mapping process to track the objects. The presented results include a nine node indoor camera network to demonstrate that this autonomous approach is feasible and offers an acceptable level of accuracy in terms of computed object locations.

1.3.5 Chapter 6: Object Association for Tracking in Non-overlapping Wireless Smart Cameras

Chapter 6 considers the problem of object tracking in a distributed wireless smart camera network. The vast majority of current object tracking techniques, either centralised or distributed, assume unlimited energy, meaning these techniques do not translate well when applied within the constraints of low-power

distributed systems. Chapter 6 presents a highly-scalable, distributed strategy to object tracking in wireless smart camera networks with limited resources. In the proposed system, cameras transmit descriptions of objects to a selected set of neighbours, determined using a predictive forwarding strategy. The received descriptions are then matched at the "next" cameras on the objects' path using a probability maximisation process with locally generated descriptions. It is shown via simulation, that the predictive forwarding and probabilistic matching strategy can significantly reduce the number of object-misses, ID-switches and ID-losses; it can also reduce the number of required transmissions over a simple broadcast scenario by up to 67%.

1.3.6 Chapter 7: Statistical Determination of Optimal Camera Configuration

Chapter 7 presents a solution to the problem of selecting optimal camera configurations (camera locations, orientations etc.) for multi-camera networks. Previous approaches largely focus on proposing various objective functions to achieve different tasks. Most of them, however, do not generalise well to large scale networks. To tackle this, a statistical formulation of the problem is introduced and a Trans-Dimensional Simulated Annealing (TDSA) algorithm is proposed to effectively solve the problem. The proposed approach is compared with a state-of-the-art method based on Binary Integer Programming (BIP), which shows that the described approach offers similar performance on small scale problems. However, the capability of the proposed approach in dealing with large scale problems is also demonstrated through comparisons with 2 alternative heuristic methods designed to address the scalability issue of BIP.

1.3.7 Chapter 8: Conclusions and Future Work

This final chapter presents the conclusions of the thesis, and considers possible future directions.

1.4 Original Contribution of the Thesis

The original contributions made in this thesis are as below:

1.4.1 Self-calibration of the Internal Parameters of Wireless Smart Cameras (Chapter 4)

- A new approach to imposing extra constraints in order to automatically calibrate cameras in degenerate configurations of pure panning or pure tilting is proposed.
- An improvement to the generic RANSAC, based on analysing the geometric distribution of feature points is reported. The improvement is capable of reducing the processing time and increasing the likelihood of finding more accurate homographies. As homography computation is a critical step in the calibration procedure, this improvement leads to significantly better calibration results.
- This work doesn't assume perfect lens (in contrast to many conventional self-calibration methods). The estimation of lens distortion (2^{nd} order radial distortion) is fully integrated into the self-calibration procedure and significantly improves the result of calibration.

This work has two resultant publications:

J. Liu, D. O'Rourke, T. Wark, R. Lakemond and S. Sridharan, "Camera calibration in wireless multimedia sensor networks," in *Proc. of the International Conference on Intelligent Sensors, Sensor Networks and Information Processing,*, Melbourne, Australia, 2009, pp. 301–306. (**Best student paper award**).

J. Liu, T. Wark, R. Lakemond and S. Sridharan, "Self-calibration of Wireless Cameras with Restricted Degrees of Freedom," *Computer Vision and Image Understanding*. (Impact factor 2.53.)

1.4.2 Autonomous Calibration of Wireless Smart Camera Networks and Object Localisation (Chapter 5)

- A novel approach is designed to calibrate a network of cameras in terms of their image plane-ground plane homographies in a non-overlapping distributed wireless camera network. The approach is fully autonomous and doesn't need any manual input as required by conventional methods. Therefore the approach is scalable and flexible.

This work has one resultant publication:

J. Liu, T. Wark, S. Martin, P. Corke and M. D'Souza, "Distributed object tracking with robot and disjoint camera networks," in *Proc. PerCom - WORKSHOPS: IEEE International Conference on Pervasive Computing and Communications Workshops*, Seattle, WA, 2011, pp. 380-383.

1.4.3 Object Association for Tracking in Non-overlapping Wireless Smart Cameras (Chapter 6)

- A network level protocol is designed for the problem of target handover in a non-overlapping distributed wireless camera networks. The designed protocol improves the accuracy of the existing master/slave mechanism and it is capable of reducing network traffic significantly.

This work has one resultant publication:

J. Liu, D. O'Rourke, T. Wark, S. Denman and S. Sridharan, "A distributed protocol for object tracking in wireless multimedia sensor networks," in *Proc. of the International Conference on Intelligent Sensors, Sensor Networks and Information Processing*, Brisbane, Australia, 2010, pp. 67–72.

1.4.4 Statistical Determination of Optimal Camera Configuration (Chapter 7)

- A generalised statistical framework is introduced for the problem of selecting the optimal camera configuration, considering a number of user constraints and unknown number of cameras. This formulation is much more flexible in incorporating various user requirements than alternative approaches.
- A Trans-Dimensional Simulated Annealing (TDSA) algorithm is proposed to solve the problem. The TDSA algorithm, based on Monte Carlo sampling, is much more efficient in terms of speed than Linear Programming based approaches and is more accurate than heuristic based approaches.

This work has one resultant publication:

J. Liu, C. Fookes, T. Wark and S. Sridharan, "On the statistical determination of optimal camera configurations in large scale surveillance networks," in *Proc. European Conference on Computer Vision - Volume Part I*, Firenze, Italy, 2012. pp. 44–57

Chapter 2

Background on Distributed Wireless Smart Camera Platforms

Distributed wireless smart cameras are real-time distributed embedded systems that perform computer vision tasks using multiple cameras. It represents a interdisciplinary field that combines techniques from computer vision, distributed and embedded computing and wireless sensor network. The fundamental role of this type of camera is no longer taking pictures but instead they also need to analyse the scene and report events of interest back to a base station or transmit the information to neighbouring cameras for collaborative signal processing tasks, such as object tracking. Transmitting all of the video feeds from a large number of wireless cameras to a central server is expensive in terms of energy and bandwidth and in fact, for most of the current wireless smart cameras, it is beyond the capability that can be handled by the slow wireless link. Furthermore, this is inherently unscalable and the combination of a large number of nodes,

fast response times, and constantly changing relationships between the cameras pushes the algorithms to be used on this platform away from server based architectures. Distributed computing provide a realistic approach to the creation of large distributed wireless smart camera systems.

With its unique benefits and challenges, DWSC is vastly different to current wired camera networks. This chapter aims to provide adequate introductory knowledge of the platform with emphasis on the benefits and challenges that significantly influenced the design of the algorithms presented in later chapters of this thesis. The introduction of the platform starts with a description of the platform that motivates this project and a number of similar ones used by other researchers in this field. This chapter ends with a discussion of the advantages and disadvantages of this type of platform.

2.1 The Motivating Platform

Commonwealth Scientific and Industrial Research Organisation of Australia (CSIRO) has been developing a Wireless Multi-media Sensor Network (WMSN) for long term outdoor environmental monitoring applications. Each device in the network, also referred to as a node, comprises a wireless module and Digital Signal Processor (DSP) for both communication and signal processing, respectively. The particular wireless module used in this project is known as the Fleck^{TM}-3z. The Fleck^{TM}platform is a robust family of devices designed at CSIRO for outdoor low-power wireless communications. The WMSN node consists of many different type of sensors, including a camera, 2 microphones and 3 passive infra-red sensors. Among these, only camera is concerned in this project and therefore the platform is effectively a wireless smart camera.

2.1.1 Digital Signal Processor

The digital signal processor is an Analog Devices 600MHz ADSP-BF537 Blackfin processor. This processor is a combination of a high-performance media processor and compiler-friendly processor optimised for both control and signal-processing operations. It operates as a 16-bit digital signal processor and a 32-bit micro-controller unit (MCU) simultaneously and supports dynamic memory access and cache memory controllers for improved efficiency. The implementation used is the BlueTechnix CM-BF537E board which comprises the CPU with 132KB internal SRAM, 32MB external SDRAM and 4MB flash.

Computationally, the Blackfin uses a fixed point architecture. In general, the cutting-edge fixed point processors tend to be faster, more power conscious and cost-sensitive while floating point processors offer high precision at a wide dynamic range. With the high speed of the Blackfin processor, it is possible to emulate floating-point operations trading off computational efficiency for low-cost and low-power operations for scenarios where floating point computation is only required on occasion. Ko et al [69] conducted a detailed comparison between fixed and floating point operations on Blackfin BF537 and showed that fixed point operations offers much faster computation for little reduction in precision.

2.1.2 Image Sensor

The image sensor used is the OV9655 (as used in CITRIC [20]). This is a 1.3 mega-pixel quarter inch sensor, producing images in a wide range of formats. It supports image sizes SXGA (1280×1024), VGA (640×480), CIF (352×288), and any size scaling down to 40×30. The image array is capable of operating at up to 30 frames per second (fps) in VGA, CIF, and lower resolutions, and 15 fps

in SXGA. The typical active power consumption is 90 mW for the OV9655 and the standby current is less than 20 μA. The choice of this image sensor is due to its balanced performance and energy consumption.

2.1.3 MicroSD Slot

The microSD slot takes up very little space on the board and with the current cards, 16GB of data can be stored.

2.1.4 Servo Mechanism

In [89], a number of potential servos that could be used with the system were analysed. The result of this analysis suggests that digital servos seems to offer the most promise in terms of overall performance. However, in terms of cost, analog servos tend to be cheaper. Two very promising servos are therefore the HS-311 (analog) and the HS-5065MG (digital). The price difference between the two is approximately $40 ($7 for the HS-311). However, the precision and speed of the HS-5065MG makes it the servo of choice for more critical applications. It also consumes less energy (approximately .2J/90 degree turn compared with .3J/90 degree turn).

2.1.5 Radio Daughter Board

Inspired by the original Berkeley mote, since 2002 CSIRO have developed a number of generations of devices known as the Fleck^{TM}. The current version, Fleck-3z, consists of an Atmega128 micro-controller running at 8 MHz, a Zigbee radio transceiver. With a 15cm whip antenna, ranging up to 500 meters in outdoor en-

vironments, the fleck can send at a data rate of 150 Kbits/s (Kbps). The Fleck-3z platform is also designed for easy expansion via add-on daughter boards which communicate via digital I/O pins, serial port and the Serial Peripheral Interface (SPI) bus. This board is responsible for wireless communication between different camera nodes.

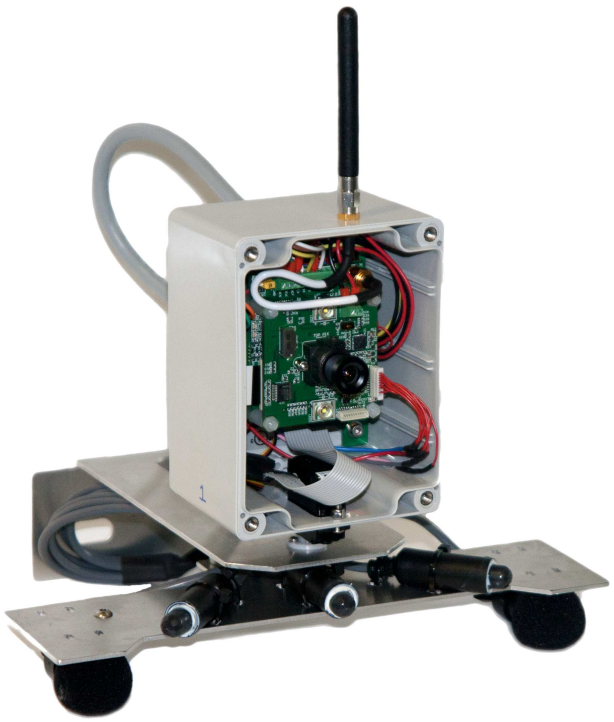


Figure 2.1: CSIRO distributed wireless smart camera platform.

2.2 Other Platforms

There are a number of platforms developed by other research groups. Each one of them is described below followed by a comparison with the hardware used in this project.

MeshEye system [53] developed by Stanford University has been known for its

unique vision system: a low-resolution stereo vision system (30 × 30 pixel, 6-grayscale) continuously determines position, range, and size of moving objects entering its field of view. The stereo vision system monitors the environment and wakes up the mote for more sophisticated processing of images captured by the other CMOS camera(640 × 480 images) in presence of motion. All processing load is handled by the ARM7TDMI ARM Thumb processor running at 55MHz.

FireFly Mosaic [102], designed by Carnegie Mellon University (CMU) is very similar to the hardware used in this project. It consists of a CMUcam3 vision processing board, a FireFly sensor networking node and a FireFly gateway to PC interface board. The CMUcam3 board features a OmniVision OV6620 capable of producing fifty 352×288 colour images per second, a 32-bit LPC2106 ARM7TDMI micro-controller running at 60MHz with 64KB of on-chip RAM and 128KB of on chip FLASH memory. The system has been used in a home activity clustering application where it automatically combines information extracted from multiple potentially overlapping cameras to recognise various regions in the house where particular activities frequently occur.

The UCLA Cyclops [84] node consists mainly of an imager, a 8-bit micro-controller (MCU), a complex programmable logic device (CPLD), a 64KB external SRAM and a 512KB external Flash. Operating at 7.37MHz and 3.3V, the micro-controller controls Cyclops and performs image processing, while the CPLD is used as a light weight frame grabber. The imager is an ultra compact CIF resolution CMOS camera module, capable of 352 × 288 images continuously. However the actual resolution is only 128 × 128 due to limited RAM and slow processing of images (1 to 2FPS). This system has been used for object tracking and hand gesture recognition.

The UCBerkley CITRIC mote [20] consists of a OV9655 imager, capable of producing VGA images at 30 fps, a PXA 270 ARM processor running at 624MHz,

	Processor	Radio	Camera	Memory
CISRO WSC	BF537 @600MHz	Fleck3z @250Kbps	640 × 480	32MB
MeshEye	Arm @55Mhz	802.15.4 @250Kbps	Two 30 × 30 One 640 × 480	64KB
FireFly Mosaic	MCU @60Mhz	802.15.4 @250Kbps	352 × 288	192KB
Cyclops	MCU@7.4Mhz CPLD@16Mhz	MICA2 @76.8Kbps	352 × 288	64KB
CITRIC	PXA270 @624Mhz	802.15.4 @250Kbps	640 × 480	64MB

Table 2.1: Comparison of existing platforms.

64MB RAM and 16Mb flash. The mote is equipped with a CC2429 radio, transmitting at 250kps. The mote has been used to perform object recognition tasks.

In [21], the authors described a video sensor platform consisting of a smart image sensor with focal plane motion processing implemented at pixel level in highly paralleled and efficient analog circuits. The camera node is intended for ultra-low bandwidth ad-hoc wireless networks with data rates of less than 2kB/sec. The CMOS image sensor (consuming 4.2mW of power at 30 frames per second, 90×90 pixels) employed autonomously monitors for scene changes, outputting full image data only when relevant. The entire sensor module can operate from 3 AA batteries and consumes 225mW at full operation. Their node will autonomously prioritise and transmit the most critical data over the low-bandwidth network.

There are a number of other platforms proposed in literatures [19, 23, 94] which will not be described here.

2.3 Distributed Wireless Smart Camera Properties and Challenges

Despite the vast application domain and the advantages over existing camera networks, the DWSC platform suffers from a number of constraints and many challenges are to be faced when designing suitable algorithms. Some of these limitations are brought by the hardware while others are due to the distributed nature of the network. In the following, a number of positive and negative features of DWSCs are described.

2.3.1 Difficulty in Video Transmission

Although transmitting video back to centralised servers is not the fundamental task of the platform, it is often needed in certain situations where human observations are required. This is extremely difficult on DWSCs due to a number of reasons. The most obvious one is the transmission rate of the wireless link which may not be able to accommodate the needs unless WiFi is used. Secondly, the channel has a certain amount of bandwidth available which does not allow a large quantity of cameras to transmit constantly. Thirdly, the amount of energy consumed is prohibitive on nodes powered by batteries. Distributing larger amounts of power requires a more substantial power distribution network which increases the installation cost of the system. The alternative is to store the video locally on each camera thanks to the recent advances in the capacity of microSD cards. However physically collecting the video remains a tedious and time consuming task.

2.3.2 Local Execution

Each smart camera is equipped with a processor of some sort that enables local processing on the device. Due to the power constraint, these processors tend to be power-efficient ones with limited processing capabilities and often run at low frequencies. For example, 55MHz in MeschEye, 60MHz in FireFly and 7.8Mhz in Cyclops, 18MHz in nodes described in [21]. This indeed reduces the power consumption of overall the system but at the same time limits the type and performance of the applications that can be developed on the platforms. Even the fastest processors such as BF537 or PXA270 are still a far cry from standard desktop PCs. Thus the node-level algorithm must not be computation intensive in order to ensure smooth execution on these devices.

2.3.3 Real-Time Requirement

Real-time requirement argues in favour of computing locally since the round-trip-delay of a wireless link is typically high. Having the cameras to compute the captured image locally eliminates this delay and thus better satisfies the real-time requirement. However, with the real-time requirement the images must be processed in a relatively faster manner, which puts a heavy burden on the processors. Not only this will increase energy consumed but further puts a restriction on the algorithms. For example, real-time tracking of SIFT features is plausible on desktop PCs but is infeasible on DWSC since it takes a few seconds to extract SIFT features from an image [69].

2.3.4 Multi-Purpose

Current wired camera infrastructures often serve multiple purposes to maximise the return of the invested capital. For example, the video feeds from multiple cameras may be simultaneously used for facial recognition and fall detection. If more tasks are added that requires extra processing capability, new servers can be purchased and wired to the existing network. Although multiple purposes may be required, the DWSC platform can only cope with them to a certain extend as addition of extra processing capability is an extremely difficult task and requires redesign, reproduction and redeployment of the cameras. The alternative is to maximise the use of existing processing capabilities. Often in a surveillance network, events or object of interests happen/appear within a few small regions comparing to the coverage of the entire of the network and therefore the processing load can be distributed to nearby idle cameras.

2.3.5 Collaborative Signal Processing

Collaborative signal processing requires data to be shared among DWSCs but not all pairs of DWSCs need to communicate with each other. If managed intelligently, information from one camera will only be transmitted to the other cameras that need it, therefore limiting the network traffic, reducing energy consumption and improving bandwidth efficiency. In addition, to process data in a distributed camera network, the cameras must be accurately calibrated and properly time synchronised (computing the spatial and temporal relations of all cameras). Thanks to the advances in wireless sensor network research, time-synchronisation is a solved topic but camera network calibration still remains an on-going research in the field of computer vision.

2.3.6 Autonomy and Adaptation

An important feature of DWSCs is that they can be easily deployed and scale to large numbers. This advantage must be preserved by the algorithms that are executed on these devices. Autonomy is thus one of the key requirement when designing the algorithms. For example, traditional cameras may be calibrated using manually marked points in the 3D space but in the domain of DWSC, this is not suitable at all due to the lack of scalability. Further, the cameras will most likely be used in dynamic environments with variable number of cameras, dynamic wireless links etc. Thus they need to self-organise and adapt to the environment without difficult human intervention.

2.3.7 User Interaction and Privacy

In future, as a result of distributed local processing the cameras may act as a passive sensor platform as well as a interactive user-centric and actuator platform. The network not only extracts information from the environment but also interacts with the user and provide feedback upon request. In addition, since rarely the video feeds will be transmitted, DWSCs are better at preserving users privacy and security.

2.4 Chapter Summary

This chapter has provided an rudimentary introduction to distributed wireless smart camera networks. It started with a description of the hardware used in this project followed by descriptions of other comparable platforms developed by other research institutes in the community. Later sections highlighted a number

of critical features of DWSCs that influence the design of algorithms for the platform. Some of these features can be considered as advantages over DWSC's counter part — the wired camera networks, while others are constraints that put limits on the capability of the platform.

Chapter 3

Background on Relevant Computer Vision Principles and Techniques

This chapter first introduces the basics of projective geometry, which encompass camera geometry and the formation of images through 3D to 2D projections. Then a number of concepts that are extensively used for camera calibration, object detection and tracking will be discussed. The subject is treated in a manner that provides the understanding and algebraic tools required to formulate and solve the problem of *camera internal parameter calibration, multi-camera calibration and multi-camera tracking of objects*. This chapter solely serves the purpose of ease understanding of later chapters of the thesis. For a comprehensive discussion of the topics, interested readers are referred to [52] for projective geometry and [126] for object tracking.

3.1 Projective Geometry

3.1.1 Homogeneous Coordinates

Homogeneous coordinates are a coordinate system used in projective geometry to represent objects such as points or planes. Compared to their Cartesian counterparts, they are in most cases simpler in representation and are more symmetric.

Given a $n-$dimensional vector $\mathbf{v} \in \mathbb{R}^n$, its homogeneous representation is a $(n+1)-$dimensional vector $\mathbf{v}_h \sim [\ \mathbf{v} \quad 1\ ]^\top$, where the $\sim$ symbol denotes equality up to a scale factor, meaning $[\ \mathbf{v} \quad 1\ ]^\top$ and $[\ k\mathbf{v} \quad k\ ]^\top$ (where k is a scalar) are equivalent.

The commonly used coordinate space in which the homogeneous point, $\mathbf{p} = [\ x \quad y \quad w\ ]^\top$ is defined is referred to as the projective 2-space, $\mathbb{P}^2$ and similarly the homogeneous point $\mathbf{p} = [\ x \quad y \quad z \quad w\ ]^\top$ resides on the projective 3-space $\mathbb{P}^3$.

The homogeneous coordinate system has a number of advantages over its Cartesian (also referred to as inhomogeneous) counterpart. First, points at infinity cannot be represented in Cartesian coordinates but they have a form of $\mathbf{p} = [\ x \quad y \quad 0\ ]^\top$ in homogeneous coordinates. Second, homogeneous coordinates allow common operations such as translation, rotation, scaling and perspective projection to be implemented as matrix operations. For example, in perspective projection, a point in $\mathbb{P}^3$ is projected onto a plane with respect to a centre of projection. In the simplest setting, assume that a point $\mathbf{p} = [\ x \quad y \quad z\ ]^\top$ is projected onto the plane $z = 1$ and the centre of project is the origin. Then the process can be described in homogeneous coordinates as,

$$
\begin{aligned}
\mathbf{p}' &= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \mathbf{p} \\
&= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \\
&= \begin{bmatrix} x \\ y \\ z \end{bmatrix}.
\end{aligned}
$$

Therefore the projected point on the $z = 1$ plane is $\mathbf{p}' = [\ x/z \quad y/z\ ]^\top$

3.1.2 Projective Transformations

This section introduces the concept of projective transformation in $\mathbb{P}^2$, also called homography, and discusses its use in computer vision.

Projective transformation exists in everyday life. For example, the transformation that maps a square onto human eyes is a projective transformation. However, when squares are viewed by human, they do not appear to preserve their original shapes in most cases. In fact, in this type of transformations, common properties of the planar objects such as angles, distances, ratios of distances do not hold. However, straightness is preserved, meaning a straight line will still be a straight line after the transformation.

Projective transformation often appears in image processing. Two images of a scene projected through the same centre of projection onto two different planes

are related by a projective transformation. This is in fact the basis for a number self-calibration methods which is discussed in Chapter 4. In addition, a planar surface projected through different view points are related by homographies. The relationship between two corresponding points ($\mathbf{x}_i$ and $\mathbf{x}_j$) on two planes that are related by a homography can be described by,

$$\mathbf{x}_i = \mathbf{H}\mathbf{x}_j. \tag{3.1}$$

A homography $\mathbf{H}$ in $\mathbb{P}^2$ is represented by a invertible 3×3 matrix with a degree of freedom of 8. A minimum of four sets of point correspondences are required between the two planes to compute the homography. The Direct Linear Transform algorithm (DLT) [51] is commonly used to compute $\mathbf{H}$. Writing the elements of $\mathbf{H}$ in vector form as $\mathbf{h} = (h_1, h_2, h_3, h_4, h_5, h_6, h_7, h_8, h_9)^\top$, Equation (3.1) for n pairs of point correspondences can be rewritten as $\mathbf{A}\mathbf{h} = \mathbf{0}$, where $\mathbf{A}$ is a $2n \times 9$ matrix. Each pair of point correspondences, $(\mathbf{x}_i, \mathbf{x}_j)$, contributes to 2 rows of $\mathbf{A}$:

$$\begin{pmatrix} x_i, y_i, 1, 0, 0, 0, -x_i x_j, -y_i x_j, -x_j \\ 0, 0, 0, x_i, y_i, 1, -x_i y_j, -y_i y_j, -y_j \end{pmatrix} \tag{3.2}$$

and for $n \geq 4$, $\mathbf{h}$ can be solved using techniques such as singular value decomposition (SVD).

3.1.3 Pinhole Camera Model and Camera Calibration

In this section, the basic principles of image formation and camera model are described, which are subsequently used in later chapters.

Image formation of a camera is the process of forming a $2-$dimensional representation of a $3-$dimensional world. The most common approach of modelling this projection that causes the dimensionality reduction is called central projection

(also called perspective projection), in which a ray from a fixed point, the centre of projection, is drawn to go through a point in the 3D scene. The intersection between this ray and the projection plane is the image point of the 3D point. This type of camera model is called a pinhole model and the centre of projection is the centre of the camera lens.

The pinhole camera model (Figure 4.1(a)) can be expressed as,

$$\mathbf{x}_{2D} = \mathbf{P}\mathbf{x}_{3D}, \tag{3.3}$$

where $\mathbf{x}_{3D} = \begin{bmatrix} x & y & z & 1 \end{bmatrix}^\top$ is a world point, in projective 3-space ($\mathbb{P}^3$). $\mathbf{x}_{2D} = \begin{bmatrix} x & y & w \end{bmatrix}^\top$ is the corresponding image point in projective 2-space ($\mathbb{P}^2$), and $\mathbf{P}$ is a 3×4 projection matrix that maps the world point to the image point. The matrix $\mathbf{P}$ may be decomposed as,

$$\mathbf{P} = \mathbf{K} \left[\begin{array}{c|c} \mathbf{R} & -\mathbf{R}\mathbf{t} \end{array} \right], \tag{3.4}$$

where $\mathbf{R}$ is a rotation matrix representing the camera orientation and $\mathbf{t}$ is a translation vector representing camera centre in the world coordinate frame. The matrix $\mathbf{K}$ has the following form,

$$\mathbf{K} = \begin{bmatrix} \gamma f_0 & s & u_0 \\ 0 & f_0 & v_0 \\ 0 & 0 & 1 \end{bmatrix}.$$

The parameters in the matrix $\mathbf{K}$ are referred to as the intrinsic camera parameters, and consist of the camera focal length f_0 (in pixel) and principal point (u_0, v_0), aspect ratio γ and skew s. For most modern cameras, including ours, $\gamma = 1$ and $s = 0$.

The external parameters of a camera refer to the orientation (θ_x, θ_y and θ_z) and position of the camera (x, y and z), which are encoded in the matrix $\mathbf{R}$ and vector $\mathbf{t}$ respectively.

Camera calibration refers to the finding of both internal and external parameters of a camera, but sometimes it also refers to the finding of a subset of these parameters.

3.2 Object Detection, Tracking and Localisation

A number of introductory concepts related to the topic of object detection, tracking and localisation are described in this section.

3.2.1 Foreground Segmentation

To detect objects, they must be firstly segmented from the rest of the image. An image can be segmented into foreground and background: Foreground contains objects of particular interest. In the case of object tracking, the moving target is of interest. On the other hand, for object recognition, the foreground may be some static portion of the scene. Since only object tracking is considered in this project, foreground segmentation is equivalent to motion segmentation, which is the segmentation of those pixels that undergo rapid change in value, as a result of motion. Although no prior information about the object of interest is available in most cases, a major part of the scene reminds constant, forming the basis of many motion segmentation algorithms. The challenges that most motion segmentation algorithms have to face are:

- Dynamic environments including small lighting variations, small background repetitive motions etc.;

- Noise caused by camera motions, which is a result of the wind in most outdoor cases;
- Real-time requirements for higher level applications such as tracking.

In this section three typical types of motion segmentation algorithm are described.

Simple Background Subtraction

The first approach is the simple background subtraction. It averages the images to create a background approximation that is similar to the current scene, except for the region where motion occurs. A new frame is subtracted from the background approximation, resulting in an difference image. This image is thresholded and outputs a motion image.

This method is effective in situations where objects move continuously and the background is visible a significant portion of the time. The method is very sensitive to noise, background motions and object speed. A predefined learning rate limits its ability to adapt to dynamic situations where objects may become stationary in a scene and start to move again. However, despite all the disadvantages, this simple method remains one of the plausible motion segmentation algorithms for embedded platforms due to its computational simplicity and reasonable good performance in stationary environments.

Variance Based

In contrast to the first approach, where pixel value is used as the measure of classification, the second approach is based on the difference between two consecutive frames (frame differencing). Typical methods of this kind include [22], in which

the authors assumed that the longer a pixel remains roughly the same, the more probable that it belongs to the background. Whether a pixel is stationary and whether it belongs to a moving object is determined by thresholding two consecutive frames. A pixel is copied to the background when its stationary count is higher than a threshold. This particular method cannot deal with background motions.

Although consecutive two-frame differencing is highly adaptive to changes in the scene, it is also very sensitive to noise. Joo and Zheng [60] use information from multiple frames to compute the variance of pixels in a recursive manner, since variance over a time interval reflects the amount of change. The equations for updating the mean and variance is,

$$\begin{aligned} m_t &= ((N-1)m_{t-1} + x_t)/N, \\ n_t &= ((N-1)n_{t-1} + x_t^2)/N, \\ v_t &= n_t - m_t^2, \end{aligned} \tag{3.5}$$

where m_t is the mean of a pixel at a particular time, v_t is the variance of the pixel and N is the pre-define number that controls the rate of learning. Note that when setting $N = 2$, this effectively reduces down to two-frame differencing.

Since variance is affected by the speed of the object, a simple background subtraction and a confidence function is integrated to suppress the trail artefact.

$$\begin{aligned} m_t &= ((N-1)m_{t-1} + x_t)/N, \\ n_t &= ((N-1)n_{t-1} + x_t^2)/N, \\ v_t &= n_t - m_t^2, \\ d &= |x_t - m_t|, \\ \sigma &= \sqrt{v_t}, \\ f_{conf} &= \begin{cases} \frac{1}{2}\left(1 - cos\left(\frac{\pi d}{r\sigma}\right)\right), & 0 \leq d \leq r\sigma \\ 1, & d \geq r\sigma, \end{cases} \end{aligned} \tag{3.6}$$

f_{conf} is very small near the background intensity so that strongly suppresses false detection caused by the trail artefact. This method is sensitive to rapid lighting changes and still leaves a trail artefact for fast moving object. Compared to the simple background subtraction, this method provides better performance at the cost of some extra processing power.

Background Modelling

The third type of motion segmentation algorithm is based on background modelling. Background modelling methods construct a model of the stationary background and then each pixel of a new frame is classified as either a foreground pixel or a background pixel. A classical algorithm of this kind is the Mixture of Gaussian (MoG) proposed by Stauffer and Grimson [107]. The idea is that the values of a pixel over time are divided into a number of layers and each layer is modelled by a Gaussian distribution with an associated weight. A new pixel is classified as foreground if it does not fit any existing background Gaussian distributions and the Gaussian with the lowest weight is replaced by a new Gaussian which takes the value of the new pixel as mean and an initial pre-defined standard deviation. However maintaining a Gaussian Mixture Model(GMM) is a heavy burden on the processor, resulting in low frame rate. This is more significant on targeted embedded processors due to limited processing capabilities. A variant of MoG is proposed in [14], where instead of using Gaussian distributions, a simpler clustering algorithm is used, achieving faster computation. This type of algorithms is robust to noise and background motion and allows the object to be stationary for a period of time before being integrated into background.

There are a number of other techniques for motion segmentation. In [90] the authors proposed a layered approach, which represents a scene as the union of pixel layers and detects foreground by propagating these layers using a maximum-

likelihood assignment. [31, 81] contain approaches based on adaptive kernel density estimation techniques. [64] presented a codebook approach, where sample background values at each pixel are quantised into codebooks that represent compressed forms of the background for a long image sequence. This method accounts for repetitive background motion and illumination variations. An embedded implementation can be found in [73]. Recently, Shen et al. developed a method that combines compressive sensing with mixture of Gaussian [105]. The approach offers similar performance to MoG but is 7 times faster, making it the idea choice for embedded platforms.

3.2.2 Object Detection

Given a motion mask, the next step is to detect objects of interest. Typically objects may be people, cars etc. One way of doing so is to use an object recognition algorithm. Object recognition algorithms generally allow different objects to be extracted from an image based on a trained model of the object, such as [121]. The advantage of this approach is that in addition to the location of the object, it also yields the type of object as the result. However, it requires pre-training and complicated classification algorithm (eg. SVM [25]) to achieve the desired accuracy and thus requiring a fast CPU and large memory that is not available on the DWSC platform in general. An alternative approach is to extract simple features out of a motion segmented image and use a heuristic based approach to isolate objects of interest. For example, a rectangle shape represents a car or a connected region greater than 500 pixel is a person. This approach is extremely fast and can be done in real-time to an acceptable level of accuracy. Object detection is needed to effectively deal with the problem of handover and occlusions. For example, Haritaoglu et al. [48] computed vertical histogram projection to locate heads and defined people based on convex hull analysis of blobs. Zhao et al. [129]

proposed an ellipsoid model in which they located heads using vertical projections and then fitted ellipses to them. Fuentes's approach [40] involves segmentation of motion images and then grouping of nearby blobs to form candidates.

Once objects of interest are detected by each camera, they can be tracked or localised using a number of techniques which are discussed below.

3.2.3 Camera Geometry in Object Tracking and Localisation

A number of geometry measures have been used by researchers in the field of video based object tracking and localisation. Tracking of objects refers to the establishment of correspondence between the observations of the same object generated by different cameras, whereas localisation means the computation of the metric location (or up to a scale factor) of the observed object.

Epipolar Geometry

Epipolar geometry has been used to establish correspondences between camera views. It is defined as the intrinsic projective geometry between two views [52] of the same scene captured by two different cameras (or the same camera at two different positions and/or orientations), which are shown in Figure 3.1.

Figure 3.1 contains a number of geometry objects:

- Baseline $\mathbf{c}_a\mathbf{c}_b$: the line connecting the two camera centres.
- Epipoles: the points of intersection of the baseline with the two image planes of the camera.

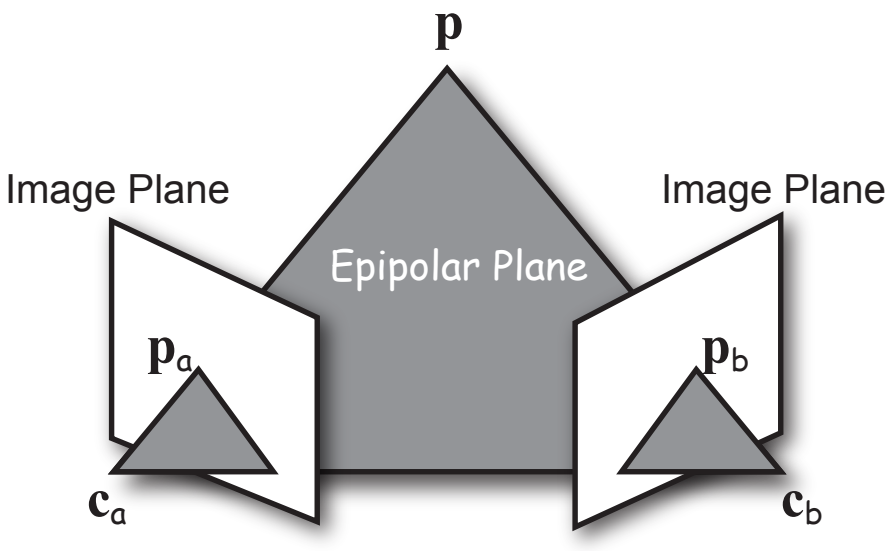


Figure 3.1: Geometry between two views of the same 3D point. $\mathbf{p}_a$ and $\mathbf{p}_b$ are projected points on the two image plane of a 3D point $\mathbf{p}$, both left and right cameras are denoted by their optical centres $\mathbf{c}_a$ and $\mathbf{c}_b$

- Epipolar planes: planes that contain the baseline, e.g. $\mathbf{c}_a\mathbf{c}_b\mathbf{p}$.
- Epipolar lines: lines of intersection between epipolar planes and the two image planes.

The epipolar geometry is encapsulated by a 3×3 matrix, which is often referred to as fundamental matrix and is assigned letter $\mathbf{F}$. As can be seen from Figure 3.1, due to the fixed structure, a point $\mathbf{p}_a$ is effective mapped to an epipolar line $\mathbf{l}_b$ on the second image. This mapping:

$$\mathbf{p}_a \longmapsto \mathbf{l}_b \tag{3.7}$$

is represented by the fundamental matrix $\mathbf{F}$ such that $\mathbf{F}\mathbf{p}_a$ describes the line on which $\mathbf{p}_b$ lies, thus $\mathbf{p}_b^\top \mathbf{F}\mathbf{p}_a = \mathbf{0}$. $\mathbf{F}$ can be derived in geometrical and algebraical fashions as described in [52] and will not be repeated here. However, it is noteworthy that only 8 (or more) feature point correspondences between the two views are required to compute $\mathbf{F}$; no calibration is needed.

Epipolar geometry can be used for associating camera observations across multiple views. Consider two images of a 3D world as shown in Figure 3.1. For the sake of simplicity, the two cameras $\mathbf{c}_a$ and $\mathbf{c}_b$ are referred to as the left and the right camera. Given a point $\mathbf{p}_a$ on the image of the left camera, if the actual 3D position of point $\mathbf{p}$ is not known, the point $\mathbf{p}_a$ can correspond to any 3D point along the line $\mathbf{p}_a\mathbf{c}_a$ in the right camera's view. In the context of multi-camera object tracking, the epipolar constraint can be used to associate objects in one view to lines in the other view, thus significantly reducing the search spaces of correspondence establishment when there are multiple simultaneously detected objects. However, as the constraint does not uniquely map points between two views, the constraint alone is sometimes insufficient. Furthermore, the constraint can only provide correspondences between two views; it is not capable of determining the object's location.

Triangulation

In many cases, knowing the correspondences of an object across multiple camera views is not sufficient; the actual locations of the object in terms of scene coordinates are also of interest. Consider the same scenario shown in Figure 3.1 in which a point $\mathbf{p}$ is projected onto the left camera $\mathbf{c}_a$ at $\mathbf{p}_a$ and onto the right camera $\mathbf{c}_b$ at $\mathbf{p}_b$. If the internal and external parameters of both cameras are available, the central projection process can be inverted such that the image point $\mathbf{p}_a$ can be mapped back to a 3D line $\mathbf{c}_a\mathbf{p}_a$ and similarly the object must also lie on the line $\mathbf{c}_b\mathbf{p}_b$. Therefore, estimating the location of the object in scene coordinates is effectively the computation of the intersection of these two lines. In a general case, each camera contributes to one of these lines and the object can be localised at the intersection of them. However, real measurements always contain errors, meaning these lines do not intersect at a single point in space. Robust estima-

tion methods such as sum of squares must be used to obtain the locations of the object [50]. It can be seen that to localise an object, at least two cameras are needed to obtain a coarse estimation. This means that a densely deployed camera network with full calibration is needed if the object is to be localised seamlessly.

Homography of Planar Scenes

The most utilised technique which allows an object's true locations to be computed is the homography of the plane on which the object moves. When it is known that the detected points on the camera views reside on a planar surface in the real world scene, the camera matrix $\mathbf{P}$ (a 3×4 matrix) can be reduced to an invertible 3×3 homography in $\mathbb{P}^2$. This assumption is approximately true in most urban environments as most human activities occur on the ground.

If writing $\mathbf{P}$ in terms of its columns: $\mathbf{P} = [\ \mathbf{p}_1 \quad \mathbf{p}_2 \quad \mathbf{p}_3 \quad \mathbf{p}_4\ ]$ and expressing the real-world 3D point $\mathbf{x}_{3D}$ as $\mathbf{x}_{3D} = [\ x \quad y \quad z \quad 1\ ]^\top$, Equation (3.3) can be rewritten as,

$$\mathbf{x}_{2D} = \begin{bmatrix} \mathbf{p}_1 & \mathbf{p}_2 & \mathbf{p}_3 & \mathbf{p}_4 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}. \tag{3.8}$$

Since the 3D world points lie on the ground, their z components are 0 and there-

fore, Equation (3.8) becomes,

$$\mathbf{x}_{2D} = \begin{bmatrix} \mathbf{p}_1 & \mathbf{p}_2 & \mathbf{p}_3 & \mathbf{p}_4 \end{bmatrix} \begin{bmatrix} x \\ y \\ 0 \\ 1 \end{bmatrix}, \tag{3.9}$$

$$\mathbf{x}_{2D} = \begin{bmatrix} \mathbf{p}_1 & \mathbf{p}_2 & \mathbf{p}_4 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}, \tag{3.10}$$

$$\mathbf{x}_{2D} = \mathbf{H}\mathbf{x}'_{3D}, \tag{3.11}$$

where $\mathbf{H} = \begin{bmatrix} \mathbf{p}_1 & \mathbf{p}_2 & \mathbf{p}_4 \end{bmatrix}$ is the 3×3 homography matrix and $\mathbf{x}_{2D}$ is the image of a ground point $\mathbf{x}_{3D}$. Planar homographies can be estimated with 4 or more pairs of point correspondences between the two planes as explained in Section 3.1.2 of this chapter.

The homograhy matrix is widely used in tracking applications [27, 28, 34, 39, 62, 63, 65, 80, 108, 112], as it can both solve the problems of associating objects across multiple camera views as well as object localisation. Because of the invertible mapping, any point on a camera view can be mapped back to an unique point on the planar surface in real world coordinates. This means that to localise an object, a *single* camera is sufficient.

To demonstrate how homography can be used to associate observations across camera views, consider the scenario shown in Figure 3.1. If denoting the homography between the plane where $\mathbf{p}$ is located and the left camera's image plane as $\mathbf{H}_a$, so that $\mathbf{p}_a = \mathbf{H}_a\mathbf{p}$, it can be seen that any point on the image can be mapped back to a point on the planar surface through $\mathbf{H}_a^{-1}$ (thus allowing localisation of the point). Furthermore, if also denoting the homography of the right camera as $\mathbf{H}_b$ and $\mathbf{p}_b = \mathbf{H}_b\mathbf{p}$, it can be easily shown that $\mathbf{p}_a = \mathbf{H}_a\mathbf{H}_b^{-1}\mathbf{p}_b$. This provides a simple way for associating observations in the right camera to that of the left

camera (i.e. tracking).

3.2.4 Multi-camera Tracking Architecture

Multi-camera tracking systems can be of many different configurations. Cameras can have either overlapping fields of view (FoVs) or they can be completely disjoint. The number of targets can vary even though recent research is more or less focusing on a single target through cameras; tracking multiple targets with satisfactory level of accuracy requires heavy computation which is not likely to be handled by wireless sensor network nodes. Most multi-camera systems are simply extensions of single camera systems [106], with trackers operating at a node level. This type of system often faces the problem of target handover. For cameras with overlapping FoVs, problems of handover and occlusions may be resolved easily if the tracker is operating above the node level. However the biggest benefit of simple extension of single camera network is scalability. Additional cameras can be added into the existing tracking framework without the need of reconfiguration. Secondly, this type of setup limits the amount of information exchange between nodes, which can be potentially problematic for wireless sensor networks due to dynamic wireless link quality and limited bandwidth. The two types of architecture are illustrated in Figure 3.2 and Figure 3.3 [29].

Figure 3.2 describes the configuration of the tracking system often used in a disjoint camera network. In this type of systems, camera nodes apply individual reasoning to track targets of interest before sharing the information with other cameras. Handover is normally achieved by sending object descriptions to their neighbouring cameras, which can then search for matchings between their own object lists and the received list. This system is widely used in camera networks in which cameras do not share common FoVs, since fusing the information

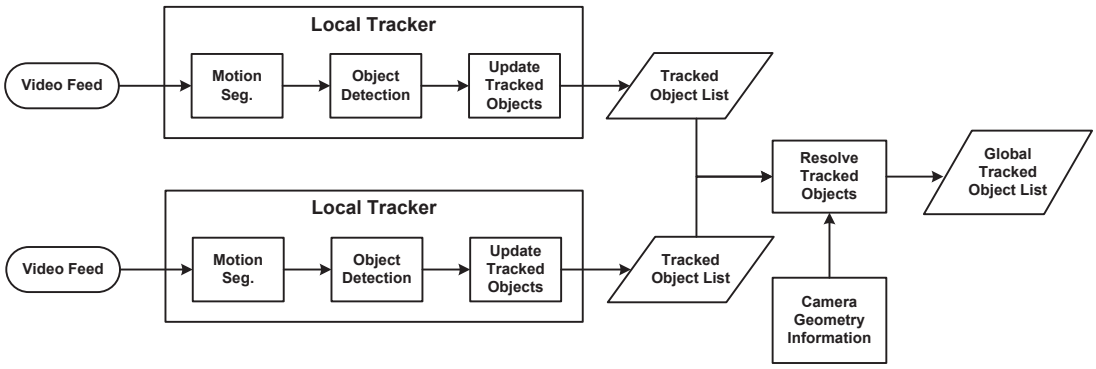


Figure 3.2: Multi-camera Tracking Architecture 1

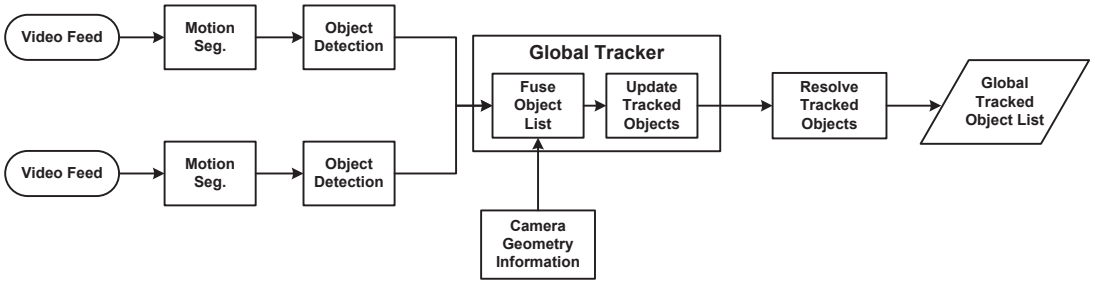


Figure 3.3: Multi-camera Tracking Architecture 2

beforehand does not bring much extra benefits, but instead, it requires more computational resources. Typical systems that implement this design can be found in [11, 68].

Systems that follow the architecture described in Figure 3.3 are often networks of cameras with overlapping FoVs. Problem of handover and occlusions in these systems can be dealt with easily. The fact that targets always exist in at least one camera has lead to the design of the architecture, which obtains a global list of objects of interest before applying any tracking algorithms. From the perspective of the tracker, which sits at a network level, the objects of interest will never be missing from its FoVs. This type of system can facilitate high precision tracking if the cameras are properly calibrated. Typical systems of this design can be

found in [79, 122].

3.3 Chapter Summary

This chapter has introduced the basics of projective geometry, which encompass camera geometry and the formation of images through 3D to 2D projections. Then a number of concepts that are extensively used for camera calibration, object detection and tracking has been discussed. The chapter solely serves the purpose of providing the understanding and algebraic tools required to formulate and solve the problem of *camera internal parameter calibration, multi-camera calibration and multi-camera tracking of objects*, which are further discussed in later chapters of this thesis.

Chapter 4

Self-calibration of the Internal Parameters of Wireless Smart Cameras

4.1 Introduction

In order for networks of cameras to be effectively used for high-level tasks such as surveillance or object detection and tracking, they may need to be calibrated. As a result of the class of camera platforms considered in this project (low cost, low-power, high numbers, distributed rather than centralised processing, etc.), a number of desirable features as well as limitations on the selection of the calibration algorithm are introduced.

Local execution - Given the reduction in cost of embedded, DSP class processors, it is assumed that camera nodes each contain sufficient processing power to be able to deal with raw image data from their own CMOS sensors. Combined with

the limitations on available radio bandwidth, this means that distributed local processing of any protocols, including calibration, is a key requirement. Manual and/or centralised procedures can be costly and impractical especially when nodes are remotely stationed or deployed in large numbers.

Efficient Algorithm - Unlike many common camera calibration procedures, which are based on the availability of desktop-grade processors, the processing power of each camera node is naturally more limited in terms of processor speed and RAM. In addition, due to the limited energy resources available at nodes, any calibration algorithm should be designed to be fast and efficient.

Camera Motion - The DWSC platform (described in Section 2.1) also has panning ability, through the use of a low-power servo mounted at the bottom of the node. This ability to pan provides a means to calibrate the camera without a known calibration object or manual intervention.

Given the requirements and constraints listed above, it is believed that an efficient automatic calibration procedure is preferable for these classes of devices. This chapter presents an efficient and effective solution to automatically calibrate a camera capable of panning or tilting motions.

The principle of the proposed approach is to make use of the panning (or tilting) ability to capture images at different camera orientations. The SURF feature extractor and descriptor [8] are employed, due to its robustness and speed, to find corresponding points between images. Since there may be a significant portion of outliers in the pool of point correspondences output by SURF, an improved version of the robust estimation method, RANdom SAmple Consensus (RANSAC) is used to compute the projective transformation between each pair of consecutive images. The intrinsic camera parameters (focal length and principal point) are computed from one or more homographies using a linear method similar to [49],

with the square pixel and known aspect ratio constraint. The angle of rotation between the images and the axis of rotation (extrinsic parameters) are also recovered.

The key contributions of the proposed approach contains:

- First, a new approach to imposing extra constraints in order to automatically calibrate cameras in degenerate configurations of pure panning or pure tilting is presented (Section 4.3).
- Second, an improvement to the generic RANSAC, based on analysing the geometric distribution of feature points is reported (Section 4.4.2). The improvement is capable of reducing the processing time and increasing the likelihood of finding more accurate homographies. As homography computation is a critical step in the calibration procedure, it will be shown that the improvement leads to significant better calibration results.
- Third, the estimation of lens distortion (2^{nd} order radial distortion) is integrated into the self-calibration procedure and the effect of radial distortion on self-calibration is demonstrated using real images.

4.2 Related Work

4.2.1 Self-calibration

Research in the field of camera self-calibration started with Faugeras et al. [35], who considered a freely moving camera with fixed internal parameters. Their technique is based on Kruppa's equations, which link the epipolar geometry to the image of the absolute conic (IAC). The IAC determines the calibration. Three

epipolar transformations, arising form three different camera motions are needed to determine the IAC uniquely (and hence the calibration), since each camera motion provides two constrains on the 5 degrees of freedom of the IAC. A number of variations of [35] can be found in [54, 74, 111].

Other well-known works in self-calibration include that of Pollefeys et al. and Triggs et al. Pollefeys [92, 93] introduced a stratified approach, which first retrieves the affine calibration of the cameras using the modulus constraint and then uses additional constraints, such as vanishing points of parallel lines in the scene, to upgrade to a Euclidean calibration. Triggs [119] imposed constraints on the camera intrinsic parameters such as constant values for focal length, skew, aspect ratio etc. in order to define equations on unknown dual absolute quadric entries. A numerical method is used to solve the set of equations, obtaining the absolute dual quadric, which may then be used to find the unknown camera intrinsic parameters.

Another class of calibration work was pioneered by Hartley [49], who imposed linear constraints on the calibration matrix using the property that the inverse of a rotation matrix is equal to the transpose of the matrix. The approach requires cameras to perform pure rotations around their optical centres. The notations of his method are described in Section 4.3.1 and will be followed in this Chapter. In [2], Agapito et al. used the mapping of the image of the absolute conic via the infinite homography to impose linear constraints on camera internal parameters. Their approach also requires the camera to undergo pure rotations at a fixed location. This method is able to determine calibration for cameras of variable focus.

One of the major drawbacks to much of the well known work in self-calibration is that they cannot deal with degenerate cases where cameras undergo pure panning or tilting movements. Zisserman et al. [131] and Sturm et al. [110] each gave a

theoretical treatment of the ambiguity inherent in various problems (including the case of panning and tilting) and Zisserman showed that it can be resolved by imposing additional constraints. However, they gave no concrete method of solving for the calibration matrix. In this chapter, this gap will be addressed by the presentation of a tractable solution to the calibration of pure panning (or tilting) cameras.

Recently Brown [13] introduced a method of computing focal length based on images that are related by panning motions. Their method uses a Maximum Likelihood Estimation SAmple Consensus (MLESAC) algorithm to compute homographies with a minimum of 2 pairs of correspondences (3 pairs if focal length is allowed to vary). Byrod [15] later extended the method to include the estimation of radial distortion. The problem with these methods is that, apart from assuming zero skew and unit aspect ratio, they also assume that the principal point is at the centre of the image, which may not be realistic for some cameras.

4.2.2 Homography Computation using RANSAC

The calibration method presented in this chapter requires one or more homographies between images taken at different camera orientations. The commonly used RANdom SAmple Consensus (RANSAC) [36] algorithm was chosen for estimating homographies [52] from a set of matched feature points between each pair of images.

The basic idea of RANSAC is to randomly select the smallest subset of data points (four pairs of correspondences in the case of homography computation), compute a model from this subset and then see how many of the available data points agree with this model. This is repeated many times in order to try and find the model that agrees with the largest number of data points. Samples are selected at

random because it is usually not feasible to evaluate every possible combination of data points. The number of trials required to ensure a good likelihood of finding the best solution depends on the number of expected outliers (and several other factors). The following formulation, proposed in [36], can be used to find the number of trials N required to achieve a probability z of obtaining the correct solution, given that p points are required per trial ($p = 4$ in this case) and the data is expected to contain w proportion inliers:

$$N = \frac{\log(1 - z)}{\log(1 - w^p)}. \tag{4.1}$$

The generic RANSAC algorithm (referred to as gRANSAC) has been modified by a number of authors in previous work. In [118], Torr and Zisserman described an extension of RANSAC. Instead of counting the number of data points which support the current hypothesis, their proposed MLESAC evaluates the likelihood of the hypothesis, representing the error distribution as a mixture model. In [77] the authors proposed R-RANSAC, which speeds up the model evaluation step by introducing a two-stage procedure. The first stage is a statistical test performed on d random samples and the second evaluation stage is carried out only if the first d data points are inliers. The idea was modified in [86], where the author included competitive verification of models. R-RANSAC was later extended in [78] to incorporate sequential probability ratio testing derived from Wald's theory of sequential decision making. Work presented by [24] achieves speed improvements to the model validation stage by introducing a pre-validation check. Similarly, [76] introduces a geometric constraint to remove outliers before applying RANSAC and [117] attempts to compute the probabilities of the validities of the correspondences and uses this information to accelerate the process.

The work that is most similar to the proposed improved RANSAC method (referred to as iRANSAC) is Zhang's bucketing scheme [127]. Zhang's method first divides the image region occupied by features into a number of rectangular buck-

ets of equal size, each of which contains a number of features. It then selects a number of different buckets based on a discrete probability distribution that is proportional to the number of features in each bucket. Once buckets have been selected, a feature is drawn randomly from each bucket. This scheme attempts to excludes features that are close to each other (because features are drawn from different buckets) but has no restrictions on the structure of features, e.g. they can be on the same line, nor the locations of the feature, e.g. they can still occupy a small region if selected buckets are clustered. Both of these scenarios lead to poorly computed homographies. The iRANSAC, presented in Section 4.4.2, addresses these shortcomings.

4.2.3 Modelling Lens Distortion

In the past, lens distortion was factored in during the non-linear refinement stage of the calibration process. This method is highly prone to error, since the initial estimate of the geometry can be poor (unless the lens distortion is very limited). In recent years, new methods have been published that allow computing the lens distortion and two-view geometry (epipolar geometry or homography) simultaneously in a linear framework. Fitzgibbon proposed a single parameter lens model, called the division model, that may be represented in a convenient form for inclusion in two-view geometry equations [37]. The problem of computing a homography or fundamental matrix and lens parameter from a minimal set of correspondences is formulated as a square quadric eigenvalue problem (QEP). Steele and Jaynes provide a method for computing the fundamental matrix in the over-constrained case, where a rectangular QEP is encountered [109]. The method to compute the homography from 5 pairs of feature correspondences is modified to allow the estimation of 2^{nd} order radial distortion to be completely integrated into the self-calibration procedure described in this chapter.

Lenses with a field of view of greater than 180° are also treated in [83]. All the above methods require only that the scene is at least partially rigid and that the lens distortion remains constant across multiple views. These conditions are met in the problems discussed in this chapter. Barreto and Daniilidis formulate the radial fundamental matrix (RFM) in [7], which allows dealing with different distortion in multiple views. It requires a minimum of 15 correspondences to compute the RFM (leading to a very costly RANSAC procedure) and is not readily applicable to computing the homography. As such, these drawbacks make it undesirable for this work.

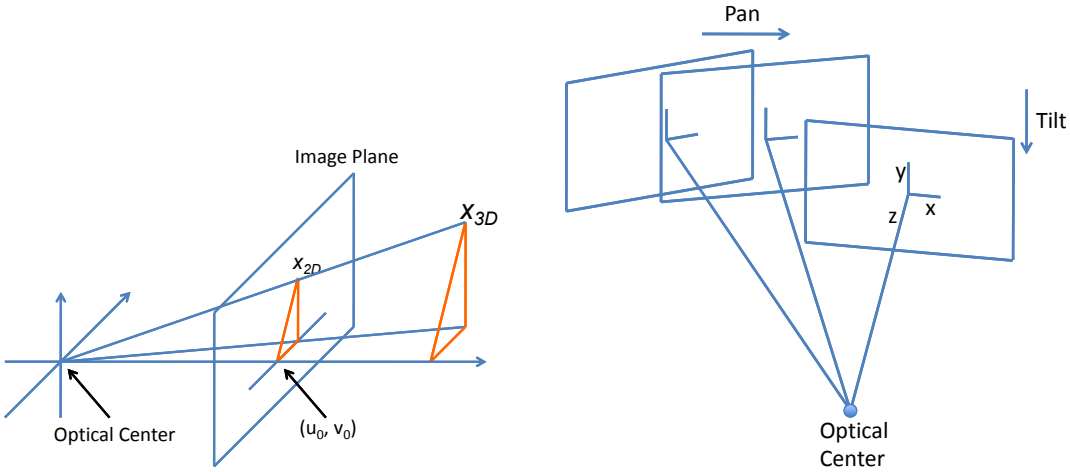
4.3 Calibrating A Panning or Tilting Camera of Fixed Intrinsics

4.3.1 Background and Notation

Camera calibration is the process of finding the parameters of a parametric model describing the camera operation. The notations used in this chapter is similar to those used in [49] (where a camera undergoing multiple rotations is treated).

During self-calibration of a stationary camera, the world coordinate system may be chosen such that the camera centre is at the origin, so that the translation vector $\mathbf{t} = \mathbf{0}$. The camera model presented in Equation (3.4) may then be simplified to $\mathbf{P} = \left[\begin{array}{c|c} \mathbf{KR} & \mathbf{0} \end{array}\right] \in \mathbb{R}^{3\times4}$. The objective is to find the unknown intrinsic calibration parameters, $\mathbf{K}$, and the extrinsic orientation, $\mathbf{R}$.

A point, $\mathbf{x}_{3D} = \begin{bmatrix} x & y & z \end{bmatrix}^\top$ (previously $\mathbf{x}_{3D} = \begin{bmatrix} x & y & z & 1 \end{bmatrix}^\top$, but the last row is omitted since the last column of $\mathbf{P}$ is $\mathbf{0}$), observed by a camera, $\mathbf{P}_i = \mathbf{KR}_i$,



(a) Pinhole camera model (b) Pan-tilt operation

Figure 4.1: Camera model and rotation

produces the imaged point, $\mathbf{x}_i = \mathbf{K}\mathbf{R}_i\mathbf{x}_{3D}$. Any two images of the same point taken by cameras $\mathbf{P}_i$ and $\mathbf{P}_j$ (two poses of the same cameras that are related by a rotation around the optical centre) are related by a homography $\mathbf{H}_{ij}$, i.e. $\mathbf{x}_i = \mathbf{H}_{ij}\mathbf{x}_j = \mathbf{K}\mathbf{R}_i\mathbf{R}_j^{-1}\mathbf{K}^{-1}\mathbf{x}_j$. If the coordinate axes are chosen to align with a reference camera, $\mathbf{P}_0$, such that $\mathbf{R}_0 = \mathbf{I}$, then the image of point $\mathbf{X}$ in any image, i, is related to the reference image according to $\mathbf{K}\mathbf{R}_i\mathbf{K}^{-1}\mathbf{x}_0$. Thus the relationship between $\mathbf{H}_i$ and $\mathbf{K}, \mathbf{R}_i$ is, $\mathbf{H}_i = \mathbf{K}\mathbf{R}_i\mathbf{K}^{-1}$, which, after rearranging gives,

$$\mathbf{R}_i = \mathbf{K}^{-1}\mathbf{H}_i\mathbf{K}. \tag{4.2}$$

Because $\mathbf{R}_i$ is a rotation matrix, $\mathbf{R}_i\mathbf{R}_i^\top = \mathbf{R}_i^\top\mathbf{R}_i = \mathbf{I}_{3\times3}$, and therefore it follows that,

$$\left(\mathbf{K}\mathbf{K}^\top\right)\mathbf{H}_i^{-\top} = \mathbf{H}_i\left(\mathbf{K}\mathbf{K}^\top\right). \tag{4.3}$$

By writing,

$$\mathbf{C} = \mathbf{K}\mathbf{K}^\top = \begin{bmatrix} u_0^2 + s^2 + \gamma^2 f_0^2 & u_0 v_0 + s f_0 & u_0 \\ u_0 v_0 + s f_0 & v_0^2 + f_0^2 & v_0 \\ u_0 & v_0 & 1 \end{bmatrix} = \begin{bmatrix} a & b & c \\ b & d & e \\ c & e & f \end{bmatrix}, \tag{4.4}$$

Equation (4.3) can be rewritten as,

$$\mathbf{C}\mathbf{H}_i^{-\top} - \mathbf{H}_i\mathbf{C} = \mathbf{0}. \tag{4.5}$$

Equation (4.5) gives rise to nine equations for the six different entries in $\mathbf{C}$. As Hartley explained in [49], at least two homographies are required for a unique solution, giving rise to an over constrained system that may be solved using least squares. Once $\mathbf{C} = \mathbf{K}\mathbf{K}^\top$ is found, $\mathbf{K}$ may be found by means of Choleski factorization and the rotation matrices may be computed using Equation (4.2).

4.3.2 The Panning Case

In this section, a new approach to imposing constraints in order to automatically calibrate cameras capable of pure panning is presented. Pure panning is a case where the camera undergoes a rotation around an axis that is parallel to the camera vertical axis and goes through the optical centre (Figure 4.1(b)). It is regarded as one of the degenerate motions under which current self-calibration algorithms fail. Among the many different types of degeneracies [6, 131], only panning and tilting motions are treated in this chapter and their causes will be briefly discussed here. Since panning is a rotation around camera y axis, it can

be described by,

$$\mathbf{R}^y = \begin{bmatrix} \cos(\theta) & 0 & \sin(\theta) \\ 0 & 1 & 0 \\ -\sin(\theta) & 0 & \cos(\theta) \end{bmatrix}.$$

Given one or more homographies, it is desirable to find the calibration matrix $\mathbf{K}$ that satisfies,

$$\mathbf{R}_i^y = \mathbf{K}^{-1}\mathbf{H}_i\mathbf{K}. \tag{4.6}$$

However there always exists a set of matrices $\mathbf{K}' = \mathbf{K}\text{diag}(1, \alpha, 1)$ that also satisfy Equation (4.6). It can be seen that the focal length in the y direction $f_0 = \mathbf{K}'(2,2)$ contains a free variable α, meaning the term is not constrained at all. Thus a constraint on the aspect ratio is required to be included in the estimation procedure, which will be discussed.

Equation (4.5) gives rise to nine linear equations for the six unique entries in $\mathbf{C}$. This system of nine equations may be expressed in matrix format as,

$$\mathbf{A}\mathbf{c}_v = \mathbf{0}, \tag{4.7}$$

where $\mathbf{c}_v = \begin{bmatrix} a & b & c & d & e & f \end{bmatrix}^\top$ is the vector of the 6 different entries of matrix $\mathbf{C}$ and $\mathbf{A}$ is the associated 9×6 coefficient matrix.

It can be shown that due to the special form of $\mathbf{K}$ and $\mathbf{R}$ and thus $\mathbf{H}$, the $\mathbf{A}$ matrices associated with pure panning motions are of rank 4 [6, 131], giving two basis vectors for the null space of $\mathbf{A}$. Since $\mathbf{c}_v$ satisfies Equation (4.7), it must be a member of the set of null space vectors of $\mathbf{A}$. If denoting the basis vectors as $\mathbf{c}_1$ and $\mathbf{c}_2$, $\mathbf{c}_v$ is then a linear combination of the two,

$$\mathbf{c}_v = \mu_1 \mathbf{c}_1 + \mu_2 \mathbf{c}_2, \tag{4.8}$$

where μ_1 and μ_2 are coefficients.

Because each of $\mathbf{c}_v$, $\mathbf{c}_1$ and $\mathbf{c}_2$ actually represents a one parameter family of vectors that are related by a scale factor (homogeneous vectors), a particular vector from each family can be used to find μ_1 and μ_2. Therefore Equation (4.8) can be rewritten as,

$$\hat{\mathbf{c}_v} = \hat{\mu}_1 \hat{\mathbf{c}}_1 + \hat{\mu}_2 \hat{\mathbf{c}}_2, \tag{4.9}$$

where

$$\begin{aligned}
\hat{\mathbf{c}}_v &= \begin{bmatrix} \hat{a} & \hat{b} & \hat{c} & \hat{d} & \hat{e} & 1 \end{bmatrix}^\top, \\
\hat{\mathbf{c}}_1 &= \begin{bmatrix} \hat{a}_1 & \hat{b}_1 & \hat{c}_1 & \hat{d}_1 & \hat{e}_1 & 1 \end{bmatrix}^\top, \\
\hat{\mathbf{c}}_2 &= \begin{bmatrix} \hat{a}_2 & \hat{b}_2 & \hat{c}_2 & \hat{d}_2 & \hat{e}_2 & 1 \end{bmatrix}^\top,
\end{aligned}$$

are simply the vectors that have 1 as the last term from their families of vectors and $\hat{\mu}_1$ and $\hat{\mu}_2$ are the coefficients associated with these vectors. To compute $\hat{\mathbf{c}}_1$ and $\hat{\mathbf{c}}_2$, singular value decomposition is used to decompose $\mathbf{A}$ into $\mathbf{A} = \mathbf{USV}^\top$, where $\mathbf{S}$ is a diagonal matrix of singular values and $\mathbf{U}$ and $\mathbf{V}$ are unitary matrices. $\mathbf{c}_1$ and $\mathbf{c}_2$ are the two columns of $\mathbf{V}$ that correspond to two zero singular values of $\mathbf{A}$. Ideally there are two zero singular values, but in the case of real images, which contain noise, there may not be any. In these cases, the two smallest singular values are used instead. $\hat{\mathbf{c}}_1$ can be found by dividing $\mathbf{c}_1$ by its last term. $\hat{\mathbf{c}}_2$ can be computed in a similar way.

Since the vectors $\hat{\mathbf{c}}_v$, $\hat{\mathbf{c}}_1$ and $\hat{\mathbf{c}}_2$ have been restricted to have 1 as their last terms, Equation (4.9) provides the following equation in $\hat{\mu}_1$ and $\hat{\mu}_2$.

$$\hat{\mu}_1 + \hat{\mu}_2 = 1 \tag{4.10}$$

Equation (4.10) alone is not enough to solve for $\hat{\mu}_1$ and $\hat{\mu}_2$; extra constraints on the entries of $\hat{\mathbf{c}}_v$ are needed. These constraints are skew, $s = 0$, and known pixel aspect ratio, i.e. γ is a known constant. From Equation (4.4) and $s = 0$, it can be deduced that,

$$\begin{aligned} \gamma^2 &= \frac{\mathbf{C}(1,1) - \mathbf{C}(1,3)^2}{\mathbf{C}(2,2) - \mathbf{C}(2,3)^2} \\ &= \frac{\hat{a} - \hat{c}^2}{\hat{d} - \hat{e}^2} \\ &= \frac{\hat{\mu}_1\hat{a}_1 + \hat{\mu}_2\hat{a}_2 - (\hat{\mu}_1\hat{c}_1 + \hat{\mu}_2\hat{c}_2)^2}{\hat{\mu}_1\hat{d}_1 + \hat{\mu}_2\hat{d}_2 - (\hat{\mu}_1\hat{e}_1 + \hat{\mu}_2\hat{e}_2)^2} \end{aligned} \tag{4.11}$$

Equation (4.10) and Equation (4.11) give rise to a system of equations (a linear equation and a quadratic equation), from which the exact values of $\hat{\mu}_1$ and $\hat{\mu}_2$ can be determined. In general, this system of equations will produce two solutions. The set of $\hat{\mu}_1$ and $\hat{\mu}_2$ that minimises the norm $||\mathbf{A}\hat{\mathbf{c}}_v||$ is selected, since in the presence of noise, the problem of finding a solution to Equation (4.7) becomes a problem of finding the vector that minimises $||\mathbf{A}\hat{\mathbf{c}}_v||$. Once $\hat{\mu}_1$ and $\hat{\mu}_2$ are found, the $\hat{\mathbf{c}}_v$ matrix can be determined using Equation (4.9), (and hence the family of vectors $\mathbf{c}_v$).

$\mathbf{K}$ may be found by applying Choleski factorization on $\mathbf{C}$ and the rotation matrices may be computed using Equation (4.2). This final step can be solved using the approach described in [49] and will not be repeated here.

In realistic scenarios, the $\mathbf{A}$ matrices are hardly of rank 4 but instead of rank 6 (i.e. full rank) due to noise from two sources — noise inherent in the feature localization process by the feature detector and the error arising from rotations not exactly around the camera optical centre. The effect of the first type of noise is treated in Section 4.5.1 and that of the second type is shown in Section 4.3.4.

4.3.3 The Tilting Case

Tilting is another type of degenerate motion in which the camera undergoes rotation along an axis parallel to the horizontal axis of the image plane. Similar to the case of panning, the motion can be represented as,

$$\mathbf{R}^x = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta) & -\sin(\theta) \\ 0 & \sin(\theta) & \cos(\theta) \end{bmatrix}.$$

Given one or more homographies, it is desirable to find the calibration matrix $\mathbf{K}$ that satisfies,

$$\mathbf{R}_i^x = \mathbf{K}^{-1}\mathbf{H}_i\mathbf{K}. \tag{4.12}$$

However, there always exists a set of matrices $\mathbf{K}' = \mathbf{K}\text{diag}(\alpha, 1, 1)$ that also satisfy Equation (4.12). In this case, the focal length in the x direction $\gamma f_0 = \mathbf{K}'(1,1)$ contains a free variable α, indicating that it is not constrained.

The proposed method naturally takes this into account, as it can also be shown that the associated $\mathbf{A}$ matrices are of rank 4, leading to the same solution as for the panning case.

4.3.4 Small Translational Offset

The calibration method presented in Section 4.3.2 requires cameras to undergo rotations that are parallel to the image x or y axis and go through the optical centre of the lens. In practice this is found hard to control as the optical centre of a lens is difficulty to estimate accurately. This type of motion is effectively a

combination of a small translation and a rotation that goes through the optical centre. It will be demonstrated that the small translation offset has very limited effect on calibration.

Representing this general motion by a projection matrix $\mathbf{P}$, which can be decomposed as described in Equation (3.4), it then follows that the image of a real world point $\mathbf{x}_{3D} = \begin{bmatrix} x & y & z & 1 \end{bmatrix}^\top$ in the projective 3-space $\mathcal{P}^3$ is $\mathbf{x} = \left[\begin{array}{c|c} \mathbf{KR} & -\mathbf{KRt} \end{array}\right] \mathbf{x}_{3D}$. Rewrite this as

$$\mathbf{x} = \begin{bmatrix} \mathbf{r}_1 \begin{bmatrix} x & y & z \end{bmatrix}^\top & -\mathbf{r}_1 \begin{bmatrix} t_1 & t_2 & t_3 \end{bmatrix}^\top \\ \mathbf{r}_2 \begin{bmatrix} x & y & z \end{bmatrix}^\top & -\mathbf{r}_2 \begin{bmatrix} t_1 & t_2 & t_3 \end{bmatrix}^\top \\ \mathbf{r}_3 \begin{bmatrix} x & y & z \end{bmatrix}^\top & -\mathbf{r}_3 \begin{bmatrix} t_1 & t_2 & t_3 \end{bmatrix}^\top \end{bmatrix}, \tag{4.13}$$

where $\mathbf{r}_i$ is the i^{th} row vector of $\mathbf{KR}$ and t_i is the i^{th} element of the translation vector $\mathbf{t}$. In practice, the z component of the real world point $\mathbf{x}_{3D}$ is usually one or more orders of magnitude larger than any component of the translational offset (e.g. a 5m distance between camera and scene compared to a 1cm $\mathbf{t}$ offset), meaning the translational vector has almost no effect on the computed image point. Because homographies are based on image point locations, therefore they are unaffected, so is the calibration.

4.4 Accurate Homography Computation

4.4.1 Computing the Homography and Lens Distortion

The calibration method presented in Section 4.3 requires one or more homographies to be calculated between images taken at different camera orientations. As shown in [115], a small lens distortion will influence the results of homography based self-calibration methods significantly. Therefore a method has been derived

to compute both homography and lens distortion parameter concurrently, which is presented in this section.

A method for computing the fundamental matrix and division model lens distortion in the overconstrained case is derived in [109]. A similar method for computing the homography is derived here which allows this stage to be fully integrated into the calibration process.

The division lens distortion model is defined as follows [37]:

$$\mathbf{p} = \mathbf{x} + \lambda \mathbf{z}, \qquad \mathbf{z} = \begin{bmatrix} 0 & 0 & r \end{bmatrix}^\top, \qquad r = x^2 + y^2, \tag{4.14}$$

with $\mathbf{p}$ the undistorted (pinhole) coordinates and $\mathbf{x} = \begin{bmatrix} x & y & 1 \end{bmatrix}^\top$ the distorted image coordinates. Using this model, the homography relation may be written in the form of a QEP as,

$$\left(\mathbf{D}_1 + \lambda \mathbf{D}_2 + \lambda^2 \mathbf{D}_3\right) \mathbf{h} = \mathbf{0}, \tag{4.15}$$

where $\mathbf{h}$ is the elements of the homography matrix in column vector form and each point correspondence, $(\mathbf{x}, \mathbf{x}')$, contributes two rows to each of (rectangular) $\mathbf{D}_1$, $\mathbf{D}_2$ and $\mathbf{D}_3$ as follows,

$$\mathbf{D}_1 = \begin{bmatrix} 0 & 0 & 0 & -x' & -y' & -1 & yx' & yy' & y \\ x' & y' & 1 & 0 & 0 & 0 & -xx' & -xy' & -x \end{bmatrix} \tag{4.16}$$

$$\mathbf{D}_2 = \begin{bmatrix} 0 & 0 & 0 & -rx' & -ry' & -r-r' & 0 & 0 & yr' \\ rx' & ry' & r+r' & 0 & 0 & 0 & 0 & 0 & -xr' \end{bmatrix} \tag{4.17}$$

$$\mathbf{D}_3 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & -rr' & 0 & 0 & 0 \\ 0 & 0 & rr' & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{4.18}$$

The rectangular QEP may be converted to a linear rectangular generalised eigenvalue problem by introducing a new variable $\mathbf{u} = \lambda \mathbf{h}$ and writing Equation (4.15) in the form [109],

$$\mathbf{D}_1 \mathbf{h} + \lambda \left(\mathbf{D}_2 \mathbf{h} + \mathbf{D}_3 \mathbf{u}\right) = \mathbf{0} \tag{4.19}$$

$$\mathbf{u} - \lambda \mathbf{h} = \mathbf{0}. \tag{4.20}$$

This can be written in the form,

$$(\mathbf{A} - \lambda\mathbf{B})\,\mathbf{v} = 0, \tag{4.21}$$

with,

$$\mathbf{A} = \begin{bmatrix} \mathbf{D}_1 & \\ & \mathbf{I}_9 \end{bmatrix}, \qquad \mathbf{B} = \begin{bmatrix} -\mathbf{D}_2 & -\mathbf{D}_3 \\ \mathbf{I}_9 & \end{bmatrix}, \qquad \mathbf{v} = \begin{bmatrix} \mathbf{h} \\ \mathbf{u} \end{bmatrix}. \tag{4.22}$$

Here $\mathbf{I}_9$ is a 9×9 identity matrix.

Equation (4.21) is solved by means of an iterative two-stage process. λ is initially set to zero (or a better estimate, if available). An update for $\mathbf{v}$ is obtained by computing the singular value decomposition of $\mathbf{A} - \lambda\mathbf{B}$ and selecting the singular vector associated with the smallest singular value. An update for λ is obtained by selecting the smallest magnitude root of the equation,

$$\mathbf{v}^\top \left(\mathbf{B}^\top + \lambda\mathbf{A}^\top\right)(\mathbf{A} - \lambda\mathbf{B})\,\mathbf{v} = 0. \tag{4.23}$$

This process converges sufficiently rapidly to be included inside the proposed iRANSAC system (Section 4.4.2) with limited modifications. Inclusion of the lens model in the homography computation brings an extra degree of freedom and the computed homography will be in terms of undistorted coordinates. In contrast to the conventional four-point homography computation algorithm [52], five point correspondences are therefore required to compute one set of homography and lens parameter.

4.4.2 Improving RANSAC

The point correspondences needed to compute homographies and lens distortion (see Section 4.4.1) are obtained by matching local image features. Local image features are patterns in an image that are distinguishable from the surrounding

image. They are matched across views by computing a descriptor for each feature from its local image neighbourhood. A comprehensive review of local image feature detectors was published in [120]. The SURF feature extractor and descriptor [8] was chosen due to its efficiency, suitability for implementation using only integer numbers and robustness for this particular task.

Depending on the number and quality of SURF features calculated, the matching process will naturally generate a significant proportion of incorrect or inaccurate matches (outliers). Too many poor matches will make techniques such as least squares estimation (LSE) ineffective for deriving the homography parameters. A much more robust method is generally required.

The RANdom SAmple Consensus (RANSAC) algorithm [36] has been shown by a number of researchers to be highly effective for the task of estimating homographies [52]. It has been observed that the generic RANSAC algorithm (gRANSAC) sometimes produces inaccurate homographies when the feature correspondences occupy only a small part of the region of overlap between images. This is primarily due to the fact that the algorithm selects four pairs from a pool of putative correspondences based on a uniform random process. Each pair of interest points has an equal chance of being selected for computing the homography. This implies that if the points are selected in a way that they are clustered (with no three co-linear) near one edge of the image, it is highly probable that points at the opposite edge are mismatched by the calculated homography.

To deal with this limitation, a modification is proposed to the randomised point selection process, indicated as iRANSAC, to improve the accuracy of RANSAC. For a set of tentative pairs, c, the variance of all points in the first image is estimated. This is done in both the x and y directions, giving σ_x^2 and σ_y^2, which indicates the spread of all the putative matches. Because inclusion of the lens model brings an extra degree of freedom, 5 pairs of correspondences are required

to estimate a homography and lens parameter. Now a set of five points can be defined as:

$$\begin{aligned}
\mathbf{p}_1 &: \ (\overline{x} - 2\sigma_x, \overline{y} - 2\sigma_y), \\
\mathbf{p}_2 &: \ (\overline{x} - 2\sigma_x, \overline{y} + 2\sigma_y), \\
\mathbf{p}_3 &: \ (\overline{x} + 2\sigma_x, \overline{y} - 2\sigma_y), \\
\mathbf{p}_4 &: \ (\overline{x} + 2\sigma_x, \overline{y} + 2\sigma_y). \\
\mathbf{p}_5 &: \ (\overline{x}, \overline{y}).
\end{aligned}$$

The first image is then divided into $s_t \times s_t$ square tiles ($s_t = 10$ in the implementation), and each feature in c is indexed according to the tile in which it falls. A tile selection process starts by sampling a random integer coordinate from a Gaussian distribution with mean at point $\mathbf{p}_i$ and x-axis standard deviation of d_x, y-axis standard deviation of d_y. d_x and d_y were chosen to be $w/8$ and $h/8$ in the implementation, where w and h are image width and height respectively. This coordinate is then converted into a tile index that indicates the tile from which a putative pair is to be selected. In case of two or more pairs existing in the same tile, a random pair is selected. If there is no putative pair in a particular tile, a random pair is selected for the closest tile that contains one or more putative pairs. The whole process is repeated for $i \in \{1, 2, 3, 4, 5\}$, selecting a set of five putative pairs to form the subset used to estimate a homography model and a lens parameter. This biases the random sample selection process towards the boundaries of the feature cluster, resulting in an increased probability of far-separated points being selected. It is noted that the improvement proposed here can be incorporated into other RANSAC variations such as MLESAC [118] etc. Figure 4.2 shows the use of a homography computed by iRANSAC in mapping one image onto another. The 5-point iRANSAC algorithm with lens distortion integrated is presented in Algorithm 4.1.

If estimation of lens parameter is not to be included in the calibration procedure, a 4-point version of iRANSAC can be used instead of the 5-point version presented earlier in this section. The difference is that the first 4 points ($\mathbf{p}_1$, $\mathbf{p}_2$, $\mathbf{p}_3$ and $\mathbf{p}_4$) are used as centres for selecting 4 input correspondences to the conventional 4-point homography computation algorithm [52].

The proposed method is different to the bucketing scheme described by Zhang [127]. Zhang's method aims to select features that are not within a certain neighbourhood (i.e. not in the same bucket), but the proposed method explicitly favours the selection of features located near the boundary of the feature cluster. When the rotation of the camera is small ($\approx 10°$, for example), features selected in this manner tend to be further away from the image centre. As a result, not only are homographies more accurately estimated, but the distortion coefficients benefit significantly as well, as the distortion effect is much more pronounced in these features and thus easier to estimate.

The proposed method incurs additional cost before the RANSAC loop to attach features to their corresponding tiles and find the closest non-empty tile for each empty tile. Compared to the computation of RANSAC loop which involves iterative estimation of homography and lens parameter, this additional cost is insignificant.

Algorithm 4.1 lists the modified (iRANSAC) algorithm. This algorithm makes use of the following subroutines:

- $(\overline{x}, \overline{y}) \leftarrow \text{MEAN}(c)$ computes the mean coordinates of the features that belong to the first image in the set of correspondences c.
- $(\sigma_x^2, \sigma_y^2) \leftarrow \text{VARIANCE}(c)$ computes the variance of coordinates of the features that belong to the first image in the set of correspondences c.

Figure 4.2: Mosaic of two images related by a rotation, circles represent matched features

- $(i_x, i_y) \leftarrow \text{RANDN}(\mathbf{x}, s_t, \sigma_x, \sigma_y)$ samples a random set of integer coordinates from a Gaussian distribution with mean at point $\mathbf{x}$ and x-axis standard deviation of σ_x, y-axis standard deviation of σ_y and zero covariance. The integer coordinates are then converted to tile indices (i_x, i_y) using tile size s_t.

- $tile \leftarrow \text{TILE}(c, s_t)$ divides the image 1 into $s_t \times s_t$ square tiles and indexes each element in c according to the tile in which the point from image 1 falls.

Algorithm 4.1: iRANSAC homography algorithm with biased sample selection.

4.1.1 Function $\{\mathrm{P}_c, c_c\} \leftarrow \mathrm{iRANSAC}\,(c, w_x, w_y)$
Input:
$c = \left\{(\mathbf{x}_1, \mathbf{x}_2)_1, (\mathbf{x}_1, \mathbf{x}_2)_2, \ldots, (\mathbf{x}_1, \mathbf{x}_2)_{n_x}\right\}$ – A set of putative correspondences, where n_x – the total number of putative pairs.
w_x, w_y – The image dimensions.
Output:
c_c – The consensus set – elements from c that all fit the selected model.
P_c – The homography fit to the consensus set using least squares.
λ_c – The estimated radial distortion parameter.
4.1.2 **begin**
4.1.3 $(\overline{x}, \overline{y}) \leftarrow \mathrm{MEAN}(c)$.
4.1.4 $(\sigma_x^2, \sigma_y^2) \leftarrow \mathrm{VARIANCE}(c)$.
4.1.5 $\mathbf{p}_1 \leftarrow (\overline{x} - 2\sigma_x, \overline{y} - 2\sigma_y)$.
4.1.6 $\mathbf{p}_2 \leftarrow (\overline{x} - 2\sigma_x, \overline{y} + 2\sigma_y)$.
4.1.7 $\mathbf{p}_3 \leftarrow (\overline{x} + 2\sigma_x, \overline{y} - 2\sigma_y)$.
4.1.8 $\mathbf{p}_4 \leftarrow (\overline{x} + 2\sigma_x, \overline{y} + 2\sigma_y)$.
4.1.9 $\mathbf{p}_5 \leftarrow (\overline{x}, \overline{y})$.
4.1.10 $tile \leftarrow \mathrm{TILTE}\,(c, s_t)$.
4.1.11 $\sigma_x \leftarrow w_x/8$.
4.1.12 $\sigma_y \leftarrow w_y/8$.
4.1.13 $c_c \leftarrow \emptyset$.
4.1.14 $n_c \leftarrow 0$.
4.1.15 $n_t \leftarrow \infty$.
4.1.16 **while** $i < n_t$ **do**
4.1.17 **for** $j \in \{1, 2, 3, 4, 5\}$ **do**
4.1.18 $(i_x, i_y) \leftarrow \mathrm{RANDN}\,(\mathbf{p}_j, s_t, \sigma_x, \sigma_y)$.
4.1.19 **if** *tile* (i_x, i_y) *contains correspondences* **then**
4.1.20 Select a random correspondence from *tile* (i_x, i_y) and store in $d_t\{j\}$.
4.1.21 **else**
4.1.22 Select a random correspondence from the closest tile that contains correspondences and store in $d_t\{j\}$.
4.1.23 **end**
4.1.24 **end**
4.1.25 Compute the trial model P_t and the lens distortion parameter λ_t from the trial set d_t.
4.1.26 Determine the consensus set for this trial c_t by checking which of c fit model P_t.
4.1.27 **if** $SIZE(c_c) > n_c$ **then**
4.1.28 $c_c \leftarrow c_t$.
4.1.29 $\lambda_c \leftarrow \lambda_t$.
4.1.30 $n_c \leftarrow \mathrm{SIZE}\,(c_t)$.
4.1.31 $w \leftarrow \mathrm{SIZE}\,(c_t)\,/\mathrm{SIZE}\,(c)$.
4.1.32 $n_t \leftarrow log(1 - 0.99)/log(1 - w^5)$.
4.1.33 **end**
4.1.34 $i \leftarrow i + 1$.
4.1.35 **end**
4.1.36 Compute the output model P_c and the lens distortion parameter λ_c using least squares fit to consensus set c_c.
4.1.37 **end**

4.5 Evaluation

4.5.1 Synthetic Data

In this section, the accuracy of the proposed calibration procedure and the improvement of the proposed iRANSAC method in comparison with gRANSAC and bRANSAC [127] is evaluated using synthetic data. A series of four simulated experiments were conducted: S1 calibration of cameras without distortion, S2 with distortion, S3 the effect of angle of rotation and S4 the effect of the number of images.

Simulated Camera - The simulated camera has a focal length of $f_0 = 1000$, pixel aspect ratio of $\gamma = 1$, skew of $s = 0$ and principal point of $(u_0, v_0) = (320, 240)$, which is the centre of the images (i.e. the size of the image is 640×480). The horizontal field of view (FoV) is approximately $2 \times \tan^{-1}(\frac{u_0}{f_0}) = 35.5°$. Depending on tests performed, the distortion coefficient used is $\lambda = 0$ or $\lambda = -3 \times 10^{-6}$.

Simulated Views - A number of simulated views were generated, with one of them being the reference view (V_{ref}) and n rotated views ($V_i, i \in \{1, 2, ..., n\}$). They formed n pairs of (V_{ref}, V_i) image pairs. In the simulation, only the panning case was considered, therefore each of the test images related the reference image by a panning motion of certain degrees. 400 features whose locations were chosen according to uniform random distributions in both x and y direction were scattered on the reference view (V_{ref}). This empirical number was based on the experience that for the SURF detector, usually a few hundreds of features can be matched across two close views. For each rotated view (V_i), the number of visible features decreases with the increase in the magnitude of rotation angle. In the simulation, although the camera has a 35.5° horizontal FoV, it was able

to capture a larger portion of the scene when a relatively large lens distortion is included(i.e. when $\lambda = -3 \times 10^{-6}$). In this case, there were still about 10 features left on the rotated view even when the angle of rotation reached approximately 45°. It is assumed that if a feature exists in the rotated view, it can be matched to the corresponding feature in the reference view. Gaussian noise of 0 mean and σ variance and 2^{nd} order radial distortion λ were also added to the feature locations; their values depend on the particular test conducted. A total number of 4 simulated tests were conducted(S1, S2, S3 and S4) and the parameters used in each test are summarised in Table 4.1.

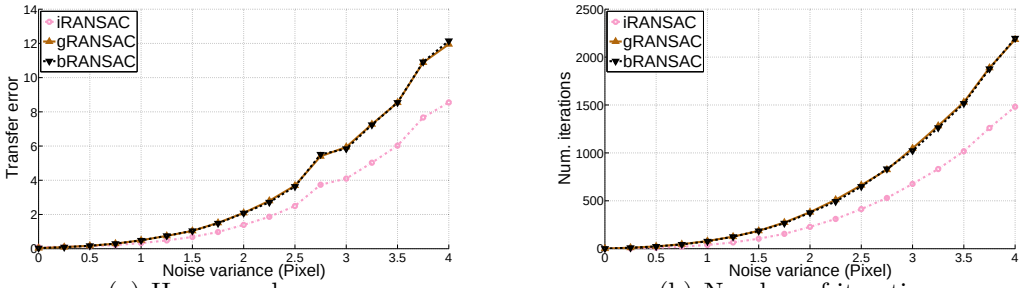
Table 4.1: Parameters used in simulation S1, S2, S3 and S4

Exp.	Est. lens	λ	Num. im	Angle	Noise
S1	No	0	5	$\pm 10°$, $\pm 5°$	0 to 4
S2	Yes	-3×10^{-6}	5	$\pm 10°$, $\pm 5°$	0 to 4
S3	Yes	-3×10^{-6}	3	$\pm 0.5°$ to $\pm 40°$	1
S4	Yes	-3×10^{-6}	2 to 11	Drawn from $\mathcal{U}(10, 15)$	1

S1. Calibration of Cameras without Lens Distortion

First, An experiment was conducted to show the performance of the proposed 4-point iRANSAC (excluding lens estimation) and the calibration method under different noise condition using the simulated camera with no lens distortion ($\lambda = 0$). In this test, the number of synthesised rotated view was $n = 4$ and they related to the reference view by pure panning of $-10°$, $-5°$, $5°$ and $10°$ respectively. For each feature noise level, whose standard deviation varied from $\sigma = 0$ to 4 with a step of 0.25, 100 independent test runs were performed.

Figure 4.3 shows the improvement of the proposed iRANSAC over gRANSAC and bRANSAC [127] in terms of symmetric transfer error per feature point and number of RANSAC trials. From Figure 4.3(a), it can be seen that the sym-

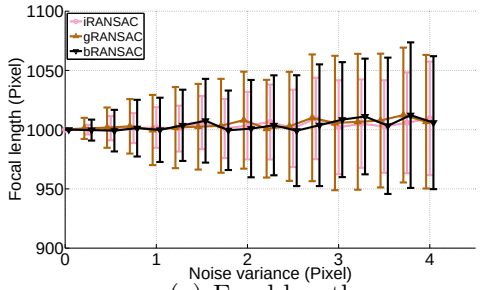


(a) Homography error. (b) Number of iterations.

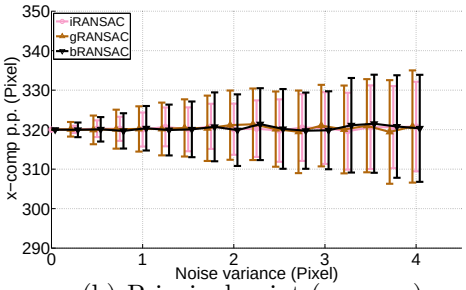
Figure 4.3: Homography error and number of iterations

metric transfer error, described in Section 4.2.2 of [52], is reduced by an average of 30.3% in all cases by iRANSAC compared to gRANSAC. iRANSAC also allows the model to be estimated within fewer iterations, as demonstrated in Figure 4.3(b) (35.7% average reduction compared to gRANSAC). The improvement in speed and accuracy is a result of the proposed random point selection scheme in iRANSAC, which produced more suitable feature correspondences for each RANSAC iteration.

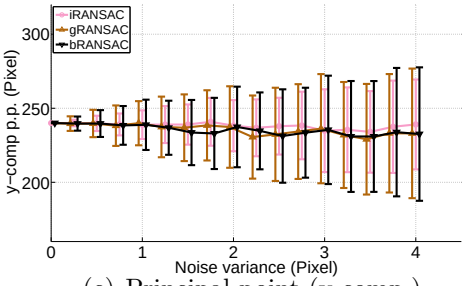
The accuracy of the calibration procedure is summarised in Figure 4.4. The ground truth values are $f_0 = 1000$, $(u_0, v_0) = (320, 240)$ for each individual plot in Figure 4.4. The standard deviation of the parameters over 100 iterations are shown as error bars on the graphs. It is noticed that almost all the mean values on Figure 4.4(a), 4.4(b) fall on the their ground truth values, indicating that calibration procedure does not introduce any bias in the estimated focal length and the x-component of the principal point. In Figure 4.4(c), it appears that there is a slight bias at high noise level for the y-component of the principal point, but it is very small compared to the mean value as well as its standard deviation. Figure 4.4(d) shows that the recovered rotation angle (only showing the angle corresponding to the first rotated view) does not contain any bias but its variation increases with the size of noise variance as expected.



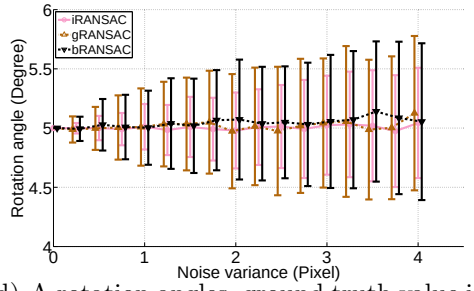
(a) Focal length.



(b) Principal point (x-comp.)



(c) Principal point (y-comp.)



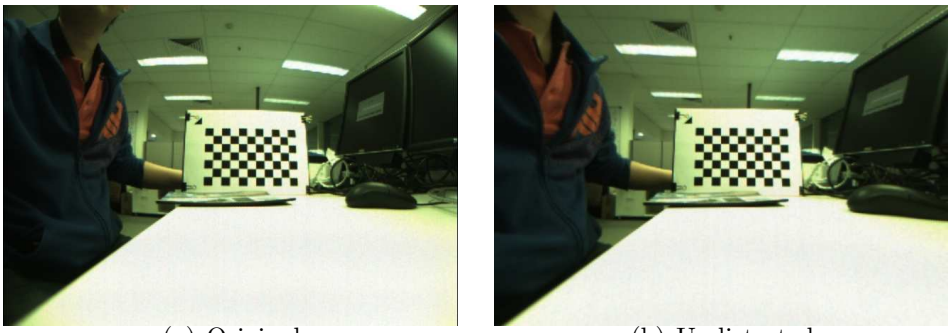
(d) A rotation angles, ground truth value is 5°.

Figure 4.4: S1 Estimated camera parameters under different noise levels. Lens distortion set to zeros.

S2. Calibration of Cameras with Lens Distortion

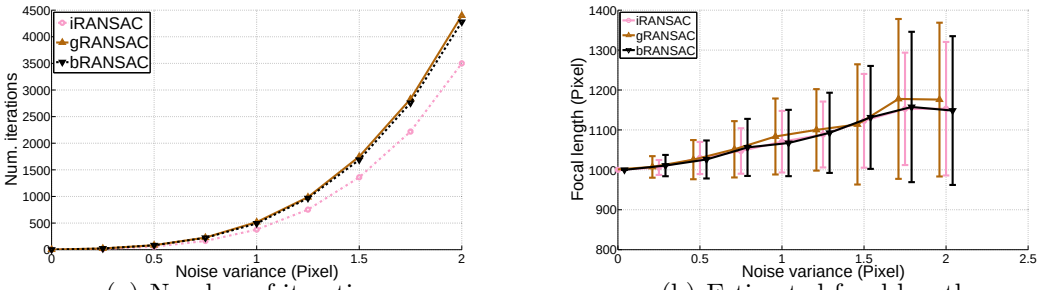
The same experiment as S1 was conducted with the exception that the lens distortion parameter was set to $\lambda = -3 \times 10^{-6}$. The visual effect of a distortion coefficient of this magnitude can be seen from Figure 4.5. Noise level only varied from $\sigma = 0$ to 2 since the results are very unstable for larger variances. The speed improvement of iRANSAC (with the estimation of lens parameter) is summarised in Figure 4.6(a) and the calibrated focal length is shown in Figure 4.6(b). Compared to the results of S1, shown in Figure 4.4, inclusion of distortion deteriorated the stability of the estimated focal length significantly and increased computation (Figure 4.6(a)). Furthermore, there appears to be a bias in the focal length which is caused by the increased level of difficulty in estimating the included distortion parameter accurately when the noise level increased. Despite the large variation

in the estimated focal length, the SURF interest point detector used can produce accurate feature locations (usually error is less than 1 pixel), therefore the proposed approach still performs well in realistic scenarios, as will be shown in Section 4.5.2.



(a) Original. (b) Undistorted.

Figure 4.5: (a) Original image captured by Videre STH-MDCS2-C camera ($\lambda \approx 3 \times 10^{-6}$) and (b) the undistorted image.



(a) Number of iterations. (b) Estimated focal length.

Figure 4.6: Number of iterations and estimated focal length. Lens distortion $\lambda = -3 \times 10^{-6}$.

S3. Magnitude of Rotation Angle

The effect of the magnitude of the rotation angle on the estimated parameter using the proposed methods was studied. In this experiment, the noise variance was set

to $\sigma = 1$ and the distortion was set to $\lambda = -3 \times 10^{-6}$. For each magnitude of the rotation angle that varied from 0.5 to 40, two rotated view ($n = 2$) were generated such that they related to the reference view by angles of the same magnitude but one is to the left and the other is to the right of the reference view. The results are presented in Figure 4.7. It can be seen that from approximately 0° to 10° the accuracy and stability increase rapidly with the increase in the magnitude of rotation angles. The cause of instability at small rotation angles is due to feature noise being dominant over the effect of the rotation. This is regarded as a near-degenerated configuration, in which the associated intrinsic parameters are poorly constrained [6]. The estimated focal length remains almost unchanged in the range from approx. 10° to 30°. As the rotation angle further increases, the number of feature pairs in the image pairs decreases. For example, at 40°, the number of pairs reduced to about 10. The influence of number of feature correspondences for homography computation is prominent when it approaches the minimum required value (5 in this case) and thus caused the homography to be poorly estimated, and hence the poorly estimated focal length.

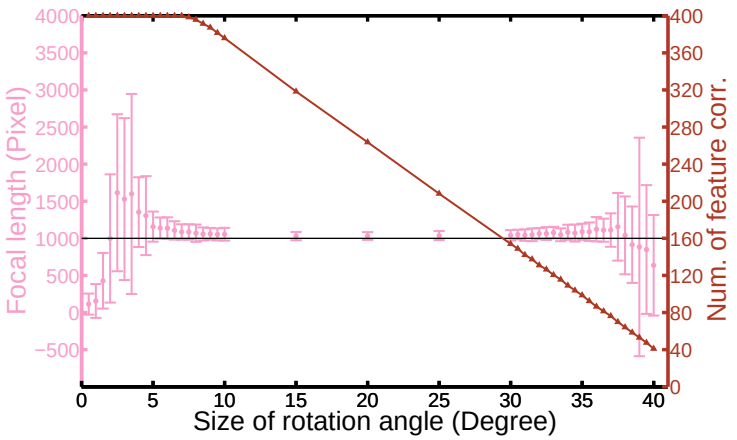


Figure 4.7: Estimated focal length and number of feature correspondences against the magnitude of rotation angle. Lens distortion $\lambda = -3 \times 10^{-6}$ and noise variance $\sigma = 1$.

S4. Number of Images

In this experiment, the number of images used to calibrate the simulated camera was allowed to vary from 2 to 11 and other parameters were kept unchanged. In each test run, the rotation angles were drawn from a uniform distribution $\mathcal{U}(10, 15)$. Noise variance and distortion coefficient were $\sigma = 1$ and $\lambda = -3 \times 10^{-6}$ respectively. The estimated focal length, plotted in Figure 4.8, shows the trend that the estimation becomes more stable as the number of image increased from 2 to 5 but having 6 or more images (i.e. 5 rotated view and a reference view) offers only limited improvement.

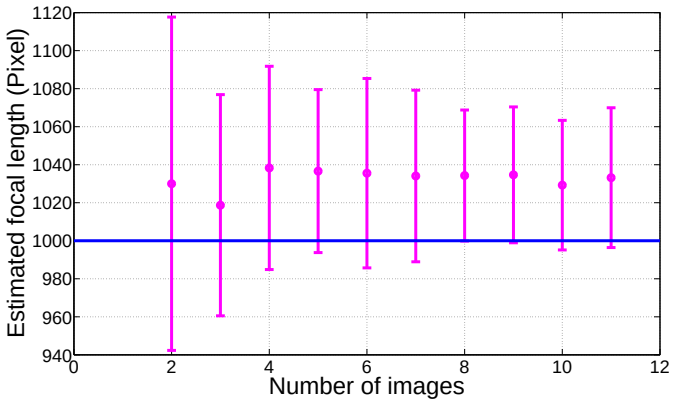


Figure 4.8: Estimated focal length against number of images. Lens distortion $\lambda = -3 \times 10^{-6}$ and noise variance $\sigma = 1$.

4.5.2 Real Data Experiment

Experimental Methodology

The motivating platform described in Section 2.1 has been chosen to experimentally validate the way the proposed procedure can be used for calibrating cameras

such as these when they are first deployed, and then periodically throughout their life if necessary. The following calibration procedure is designed to recover both the intrinsic parameters (focal length, principal point) and extrinsic parameters (angle of rotation) as well as the lens distortion parameter of the camera. However, in order to demonstrate the capability of dealing with cameras of larger distortion than that exists in the motivating platform, the Videre STH-MDCS2-C camera has also been chosen for evaluation.

Based on the methods described in this chapter, the following sequence of steps take place in the implementation:

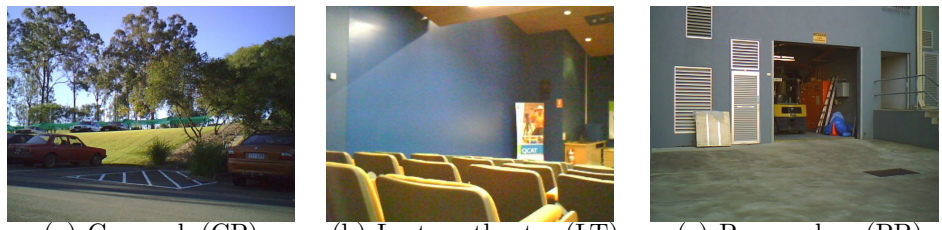
1. Capture two or more images that are related by pure panning (or tilting) motions using servo-mechanism.
2. Extract SURF features from all images.
3. Match features extracted between successive images using Euclidean distance between feature vectors.
4. Compute at least one homography and lens distortion parameter (if needed) from one or more sets of feature correspondences using the iRANSAC as presented in Section 4.4.
5. Calculate the focal length and principal point of the camera as described in Section 4.3.
6. Recover rotation matrices and subsequently rotation axis and angle, using Equation (4.2).

OV9655 Camera

A number of real image sequences (VGA quality), captured at various locations, were used to evaluate the calibration accuracy and stability of the proposed approach. The sequences that contain images related by pure panning are labelled as *car park (CP)*, *lecture theatre (LT)*, *process bay (PB)*, *office (OF)* and *driveway (DW)*. Two other sequence of images were also used, namely *PTR* and *TILT*. *PTR* is comprised of images related by rotations whose rotation axes contain x, y and z components. The use of the this sequence is to demonstrate that the proposed calibration algorithm can be used for the general case as well as for comparison with Hartley's approach. *TILT* consists of images related by tilting movements and is used to demonstrate the capability of the proposed methods in dealing with tilting motions.

Typical images from these sets are shown in Figure 4.9. It can be seen that these images contain almost no observable radial distortion, however as pointed out by Tordoff [116], a small amount of radial distortion can cause serious problems for self-calibration methods based on rotation and the infinite homography. In order to evaluate the proposed approach under these conditions, three distinct cases were tested: 1. radial distortion omitted, 2. radial distortion corrected offline, and 3. radial distortion estimated online using the approach described in Section 4.4.2. Furthermore, the accuracy of the proposed calibration procedure is also compared against the algorithm proposed by Hartley [49] using the *PTR* sequence. Note that the key benefit of the proposed algorithm is the capability of dealing with degeneracy while concurrently removing the effect of lens distortion. As such, the calibration procedure (without the removal of radial distortion component) in itself is not expected to perform significantly better or worse than Hartley's approach. Finally, although a minimum set of two images are required to calibrate a pure panning camera, three consecutive images in each of the experiments were

used to increase redundancy in the overall system.



(a) Car park (CP) (b) Lecture theater (LT) (c) Process bay (PB)

(d) Office (OF) (e) Drive way (DW) (f) PTR

Figure 4.9: Sample images.

Case 1 - Radial distortion omitted. The camera parameters (focal length and principal point) were recovered from each of the data sets by following the procedures outlined in Section 4.5.2. Because radial distortion is omitted, the 4-point versions of iRANSAC, gRANSAC and bRANSAC were used to compute the homography. The results of this experiment are summarised in Table 4.2. Mean and standard deviation of each parameter over 100 identical test runs were calculated and reported. The last entry of Table 4.2 is an offline calibration result, computed using the toolbox provided by CalTech[1] with 16 images of a standard checkerboard and manual input. The ground-truth for the internal parameters were unable to be obtained; the best achievable values are the manually calibrated ones.

Case 2 - Radial distortion corrected offline. The results of the experiment with radial distortion corrected offline are reported in Table 4.3. The lens parameter was estimated from 178 pairs of consecutive images from the *PB* and *DW* data

[1]Availabe at: http://www.vision.caltech.edu/bouguetj/calib_doc/ [accessed 01Aug2010]

sets. Images were corrected before being used to test the calibration procedure. It is clearly observable that the focal lengths estimated are very close to the that of the manual method which is the last entry of Table 4.3.

Case 3 - Radial distortion estimated online. Table 4.4 shows the results that were obtained from using the method described in Section 4.4.2 to remove radial distortion and estimate homography simultaneously. Note that, different to the previous two experiments, the 5-point version of iRANSAC, gRANSAC and bRANSAC were used instead of their 4-point versions to account for the extra degree of freedom introduced by lens distortion.

Videre STH-MDCS2-C

Apart from using the OV9655 camera with a small lens distortion, the proposed methods were also validated using Videre STH-MDCS2-C, which has a much stronger and noticeable lens distortion, as seen from Figure 4.5. The same test as carried out in **Case 3** for the OV9655 camera (Section 4.5.2) was repeated on images captured by manually panning this camera. It is noticed that as a result of the strong lens distortion, the lens parameter occasionally failed to converge to the correct value, leading to completely meaningless homographies and thus calibration results with all three methods (iRANSAC, gRANSAC and bRANSAC). Therefore, instead of reporting the mean and standard deviation, estimated parameters are shown in Table 4.5 in terms of the median, distance of median to 25% and 75% quantiles.

Results and Discussion

It can be seen from Table 4.2 that the focal length and the principal point are consistent throughout all the data sets. The principal point matches the manual calibration result but the focal length is roughly 10% higher. This consistent bias is caused by the radial distortion of the lens which was not modelled in the experiment. This bias does not exist in the results shown in Table 4.3, which indicates that if images with no radial distortion (corrected offline beforehand) are used as input to the calibration procedure, the focal length and principal point match the offline calibration results. The results presented in Table 4.4 are less accurate than that in Table 4.2 but more accurate than that in Table 4.3. It has been shown that taking the median lens distortion estimated using a large number of images will in general lead to more accurate and stable results [71]. Therefore the online approach, where lens distortion is estimated based on each pair of images, is not as accurate as the offline estimated lens parameter used to correct the images in Case 2 (which is the median lens distortion estimated using 178 images). The result for the *LT* sequence presented in Table 4.4 for Case 3 exhibits much larger standard deviation that is not consistent with all the other sequences. From inspecting the image shown in Figure 4.9(b), it can be seen images in this sequence consist mostly walls which are feature-less. Despite this, iRANSAC still managed to produce a reasonable result but not gRANSAC nor bRANSAC. This demonstrated the superiority of the proposed iRANSAC in dealing with poorly selected calibration images. For all three cases, results computed using the *TILT* data sequence do not show any difference to those computed using other sequences, indicating both panning and tilting motions are equally treated by the proposed approach. Furthermore, the accuracy of the proposed approach is compared to Hartley's approach using the *PTR* data set in all three cases. It can be seen the results produced by both approaches are almost the same, indicating that both procedures have similar level of accuracy.

Table 4.2: Calibration results for Case 1. Radial distortion not estimated – OV9655 camera.

Loc.	Motion	Alg.	f	std f	p.p.	std p.p.
CP	pan	iR+proposed	821.8	7.5	(320.7, 205.3)	(1.7, 3.3)
		gR+proposed	816.3	17.9	(317.6, 208.6)	(3.8, 5.7)
		bR+proposed	818.6	16.1	(317.9, 208.0)	(3.7, 4.9)
LT	pan	iR+proposed	826.7	9.1	(320.7, 218.5)	(2.1, 9.8)
		gR+proposed	834.1	19.7	(317.0, 207.7)	(7.9, 28.5)
		bR+proposed	837.6	23.5	(316.6, 212.1)	(8.3, 24.8)
PB	pan	iR+proposed	804.1	7.5	(322.0, 229.1)	(1.2, 5.7)
		gR+proposed	820.2	17.2	(321.9, 228.0)	(2.5, 9.4)
		bR+proposed	815.2	14.1	(321.5, 226.1)	(2.1, 9.7)
DW	pan	iR+proposed	815.6	3.0	(320.7, 230.1)	(0.8, 2.0)
		gR+proposed	812.5	6.4	(320.4, 229.7)	(1.4, 4.0)
		bR+proposed	811.9	6.7	(320.4, 229.8)	(1.5, 4.1)
OF	pan	iR+proposed	825.5	4.2	(320.7, 229.5)	(2.1, 5.6)
		gR+proposed	824.8	10.8	(317.7, 225.5)	(4.5, 15.1)
		bR+proposed	824.6	10.0	(317.9, 227.0)	(4.6, 14.7)
TILT	tilt	iR+proposed	786.9	1.8	(327.5, 228.2)	(0.5, 0.2)
		gR+proposed	781.3	11.9	(326.8, 229.2)	(4.1 0.8)
		bR+proposed	781.2	8.9	(325.5, 229.1)	(3.0 0.8)
PTR	pan,tilt,roll	iR+proposed	823.2	8.4	(335.1, 230.0)	(5.5, 6.8)
		iR+hartley	821.3	14.9	(335.0, 231.3)	(6.1, 8.7)
Manual			775	na	(326.3, 230.7)	na

For each of the three cases for the OV9655 camera, the means and standard deviations of focal length from the results calculated using the first 5 data sets were combined (*CP*, *LT*, *PB*, *DW* and *OF*) and reported in Figure 4.10. It can be seen from Figure 4.10 that incorporating radial distortion significantly improves the accuracy of the calibration, which is reflected through the means of the focal length. However, the standard deviation of the results of Case 3 is evidently larger than that of Case 1 and Case 2. This is because the online estimation of lens parameter introduces instability in the homography computation. Furthermore, it can be seen that in all cases, results generated with iRANSAC are superior than that of gRANSAC or bRANSAC in both accuracy and stability.

Results shown in Table 4.5 for the Videre STH-MDCS2-C camera are for the case when lens distortion and homographies are estimated simultaneously. Although

Table 4.3: Calibration results for Case 2. Radial distortion estimated offline – OV9655 camera.

Loc.	Motion	Alg.	f	std f	p.p.	std p.p.
CP	pan	iR+proposed	778.2	6.86	(317.3, 217.2)	(1.0, 3.4)
		gR+proposed	768.9	15.84	(318.7, 217.0)	(3.4, 5.9)
		bR+proposed	767.7	13.4	(318.1, 217.0)	(2.9, 5.1)
LT	pan	iR+proposed	774.9	8.4	(320.5, 220.7)	(2.3, 8.0)
		gR+proposed	774.6	15.8	(318.4, 213.1)	(7.8, 24.1)
		bR+proposed	774.0	17.9	(319.2, 212.3)	(7.7, 20.6)
PB	pan	iR+proposed	776.6	5.2	(321.8, 232.9)	(0.9, 4.6)
		gR+proposed	771.7	13.4	(322.4, 234.3)	(2.4, 8.3)
		bR+proposed	770.9	12.5	(322.9, 234.2)	(2.4, 7.9)
DW	pan	iR+proposed	771.5	3.1	(321.4, 235.7)	(0.7, 2.6)
		gR+proposed	769.2	5.8	(32.7, 234.6)	(1.0, 3.4)
		bR+proposed	770.2	5.8	(320.7, 234.8)	(1.1, 3.2)
OF	pan	iR+proposed	783.9	3.6	(320.6, 232.8)	(1.2, 3.3)
		gR+proposed	775.7	10.8	(317.2, 224.2)	(3.7, 17.1)
		bR+proposed	778.2	10.0	(316.9, 223.4)	(3.4, 12.6)
TILT	tilt	iR+proposed	765.9	0.5	(326.7, 229.2)	(0.2, 0.2)
		gR+proposed	767.6	5.2	(328.2, 228.9)	(1.0 0.5)
		bR+proposed	767.4	5.2	(328.2, 228.9)	(1.4 0.5)
PTR	pan tilt roll	iR+proposed	771.4	10.3	(330.3, 227.8)	(8.2, 10.8)
		iR+hartley	762.9	14.3	(330.0, 233.1)	(7.6, 10.3.3)
Manual			775	na	(326.3, 230.7)	na

during the 100 test runs, there were circumstances when the calibration failed due to lens parameter converging to wrong values, iRANSAC still managed to maintain stable and accurate results, as indicated by the median values and the small deviations of 25% quantile and 75% from the median values. In contrast, the results produced by gRANSAC are accurate as they close to the manually calibration values but very unstable in both focal length and v_0. bRANSAC's performance lies in the middle of the two – the same as the tests performed for the OV9655 camera. As mentioned in Section 4.4.2 the reason why iRANSAC performed better is that it favours the selection of features that have larger distortion component that allow easier estimation of the distortion coefficient. Another way to improve the stability of results is to use more than 5 pairs of feature correspondences (e.g. 8 pairs) to estimate lens parameter and homographies [71].

Table 4.4: Calibration results for Case 3. Radial distortion estimated online – OV9655 camera.

Loc.	Motion	Alg.	f	std f	p.p.	std p.p.
CP	pan	iR+proposed	761.2	8.0	(318.6, 222.0)	(0.9, 4.6)
		gR+proposed	753.9	19.1	(319.3, 222.0)	(2.1, 5.4)
		bR+proposed	756.4	21.0	(319.7, 221.8)	(2.9, 6.4)
LT	pan	iR+proposed	802.2	25.0	(321.0, 219.3)	(2.3, 9.3)
		gR+proposed	833.2	123.9	(321.1, 226.6)	(29.0, 140.2)
		bR+proposed	816.6	45.7	(319.3, 210.1)	(4.8, 37.5)
PB	pan	iR+proposed	707.0	13.4	(320.9, 239.5)	(0.4, 2.9)
		gR+proposed	713.6	32.7	(321.7, 235.9)	(1.3, 7.7)
		bR+proposed	705.6	30.0	(321.5, 236.2)	(1.2, 9.4)
DW	pan	iR+proposed	788.4	5.6	(320.6, 232.2)	(0.6, 1.7)
		gR+proposed	789.2	12.5	(320.5, 231.8)	(1.0, 2.9)
		bR+proposed	787.7	11.4	(320.3, 232.3)	(0.9, 2.9)
OF	pan	iR+proposed	779.6	8.1	(320.1, 231.9)	(1.3, 3.4)
		gR+proposed	790.7	26.4	(317.9, 226.3)	(3.2, 11.5)
		bR+proposed	787.1	24.5	(318.2, 228.2)	(3.2, 9.6)
Tilt	tilt	iR+proposed	769.7	0.3	(329.1, 226.2)	(0.2, 0.2)
		gR+proposed	749.7	13.1	(329.1, 226.9)	(2.6 1.2)
		bR+proposed	748.9	16.6	(328.3, 226.7)	(2.4 1.2)
PTR	pan tilt roll	iR+proposed	784.9	22.6	(336.1, 227.1)	(9.7, 12.0)
		iR+hartley	786.2	24.1	(336.5, 226.7)	(9.4, 11.6)
Manual			775	na	(326.3, 230.7)	na

Table 4.5: Calibration results, lens distortion estimated – Videre STH-MDCS2-C camera. Results shown are median values and distances from median to 25% and 75% quantiles

Motion	Alg.	med.f(25%, 75%)	med.u_0(25%, 75%)	med.v_0 (25%, 75%)
pan	iR+proposed	368.1 (13.7, 14.8)	312.0(2.6, 2.7)	219.4(10.2, 12.0)
	gR+proposed	360.3 (73.6, 28.7)	312.4(5.6, 5.0)	230.5(18.4, 120.2)
	bR+proposed	359.0 (55.1, 22.2)	311.1(4.7, 3.9)	228.6(9.4, 68.6)
	manual	356.8	329.6	240.7

4.6 Chapter Summary

Self-calibration of cameras is a crucial task that must be undertaken if the full capabilities of wireless camera networks are to be utilised for tasks such as tracking and image registration. In this chapter, a new approach to camera calibration has been presented which imposes extra constraints in order to automatically

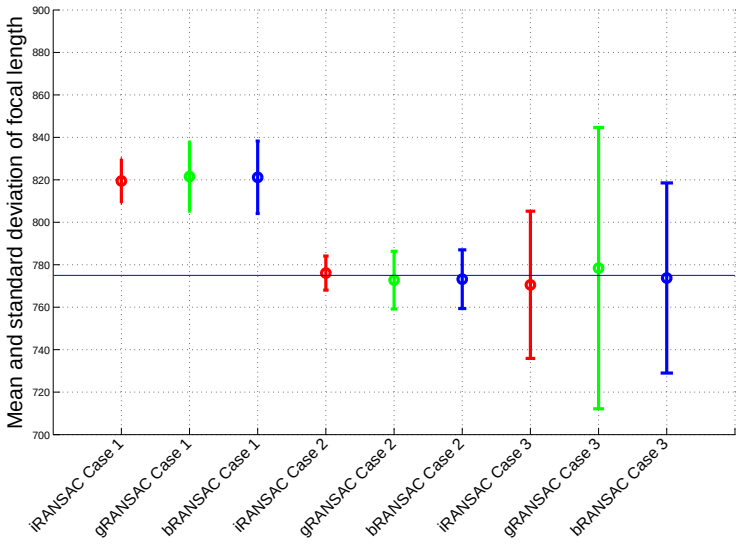


Figure 4.10: Averaged focal length over 5 data sets (100 test runs for each set) including *CP*, *LT*, *PB*, *DW* and *OF* for the 3 cases, which are radial distortion omitted, radial distortion corrected offline and radial distortion estimated online. The horizontal line represents ground truth obtained with manual calibration

calibrate cameras in the degenerate configurations of pure panning or pure tilting with fixed intrinsic parameters, recovering the focal length, principal point and angle of rotation. When the configuration is not degenerate, that is, rotations are around axes with non-zero x, y, z components, it has been shown that the proposed method has similar accuracy and stability as Hartley's approach.

To improve accuracy, an improved random sample consensus method has been introduced which is based on a biased random sample selection process. The results show that if the random sample selection process of RANSAC is biased towards selecting far-separated features, a significant increase in model accuracy and reduction in processing time can be achieved. Evaluations using both synthetic data and real data indicate that the proposed calibration procedure is accurate, stable and consistent with the ground truth values.

Further evaluation of the proposed approach has been conducted which incorporates the estimation of radial distortion in the calibration procedure. The results show that with the proposed online approach, it is possible to estimate the camera parameters more accurately than the approach where radial distortion is completely ignored. Comparing to the approach where the distortion is corrected offline with a large number of images, the proposed approach is slightly less accurate and less stable but it is much more efficient and allows implementation on autonomous systems. Thus, objective 1 in Section 1.2.2 has been fully satisfied.

Although the current version of the motivating hardware platform does not feature an automatic zoom functionality, estimating the parameters of cameras with zoom functionality is left as part of the future work. Furthermore, as noted in [131], most self-calibration algorithms do not contain estimations of error. Without error bounds, reliability of the estimated parameters may be questionable. The alternative is to run multiple iterations of the calibration procedure, which is undesirable. The estimation of error bounds is a possible direction of future research.

Chapter 5

Autonomous Calibration of Wireless Smart Camera Networks and Object Localisation

5.1 Introduction

Object localisation is a fundamental task in various video-based applications, especially surveillance. Because the limited field of view, detection and localisation is a challenging task for single cameras. Therefore multiple cameras must be used to expand the object observability. Multi-camera object localisation aims to estimate the object location across multiple camera views (overlapping and/or non-overlapping) at any given time instant.

However, to use camera networks in real environments for object localisation,

only knowing the internal parameters of the cameras is insufficient (Chapter 4); the relationship between cameras and the environment must be established. This calibration process estimates the parameters that describe the projection of 3D points with respect to a common world coordinate frame to 2D points on the image plane of each camera, allowing outputs from different cameras to share a common reference frame for making collective decisions.

With sufficiently high camera density the cameras' Field of View (FoV) will be overlapping. If the overlapping areas are large enough, corresponding points can be detected from both images from which the relative pose of cameras can be estimated with known or pre-calibrated internal parameters. Triangulation described in Section 3.2.3 can then be used to determine the object locations from two or more overlapping cameras. However, when the camera density is low, the FoV of the cameras may not necessarily overlap or when there are insufficient number of feature correspondences between the two camera views, spatial relationships between cameras cannot be obtained directly. In this chapter, both of these types of camera networks are referred to as non-overlapping camera networks for clarity of presentation.

Most current approaches to calibration of non-overlapping camera networks involve human assistance. However, for the DWSC platform considered in this project, human assisted methods are not viable since they can not maintain the highly scalability (tens to hundreds) and flexibility (easy to re-deploy) of the platform. Thus a fully autonomous approach to calibration is greatly demanded.

This chapter addresses the problem of calibrating a distributed wireless smart camera network covering a large area, where the FoV of cameras may not overlap — disjoint cameras. With the aid of a low cost mobile robot cooperating with the wireless smart camera network, it will be shown that disjoint cameras can be calibrated autonomously in terms of their homographies with respect to a

common ground plane. The resulting calibrations are used to effectively track objects moving through the network which is important in performing IVS over a wide area. The proposed approach demonstrates the effective integration of computer vision, wireless sensor networks and robotics — a cyberphysical system — to achieve the desired outcome.

5.2 Background and Related Work

Calibration of camera networks allows outputs from different cameras to share a common reference frame for data aggregation and decision making. A camera has two sets of parameters. The intrinsic parameters such as focal length, aspect ratio, skew and principal point are often assumed known apriori or estimated using methods such as the one described in Chapter 4. The extrinsic parameters describe the pose of the camera in the world in terms of its position (x, y, z) and orientation $(\theta_x, \theta_y, \theta_z)$ and these need to be estimated as well. In this section, some of the most popular techniques to calibrate camera networks will be reviewed. These techniques in general can be grouped into those for overlapping cameras and non-overlapping cameras. As the DWSC network considered in most cases belong to the latter case, techniques for calibrating non-overlapping camera networks will be the focus of discussion.

For a non-overlapping camera network, corresponding points between cameras cannot be established directly. Several researchers have proposed to use objects to counter this problem. For example, Rahimi et al. [96] demonstrated that prior knowledge about the target's dynamics can compensate for the lack of overlap by recovering the trajectory as well as the external parameters of each camera using a centralised Maximum A-Posteriori (MAP) estimation. Their approach, however, suffers from two major problems. First, the cameras are assumed to be

mounted with their optical axes normal to the ground plane in order to reduce the number of unknown parameters. Second, the performance of the system is subject to how well the target follows the estimated motion; sharp manoeuvres in the unobservable regions may cause large errors. Similar approaches can be found in [5, 41, 91]

In [72], Liu et al. proposed to use a manually placed calibration device to generate at least four reference points for each camera, allowing them to estimate their locations and orientations. However, in order to obtain 3D world positions of the calibration device, it requires an ultrasound-based sensor network to be deployed in the environment prior to calibration, which is costly and time consuming and therefore can not be generalised to networks covering large areas.

In areas where objects move on a planner (or approximately planner) surface, including most man-made environments, calibration can be reduced to computing the homography of the ground plane. This reduced form of calibration is highly popular in state-of-the-art tracking approaches [27, 28, 34, 39, 62, 63, 80, 108, 112] since it offers a compact yet effective way to convert image plane points to real-world points. To obtain such calibration, Bose and Grimson [10] used a fully automated technique for both affine and metric rectification of this ground plane (up to a scale factor) by simply tracking moving objects. A similar approach can be found in [130]. The problem with these approaches is that either human assistance is required or the scene has to contain specific geometric structures such as parallel lines or known angles.

In [98], a robot equipped with a special planar pattern collaborates with the cameras in the environment. The cameras calculate their external parameters by communicating tracking information for the planar pattern. The drawback of this approach is that either high resolution images are required, or the range of each sensor must be limited in order to correctly recognise the pattern on the

robot. High resolution images prohibit real-time onboard processing and short range means more cameras are needed to cover the same area.

The approach described in this chapter overcomes many of the short falls of these approaches [10, 72, 96, 98, 130]. Despite the use of an off-the-shelf robot, no modification of the environment (as in [72]) is required, and the process is completely automated, unlike that proposed in [10, 130]. The autonomy is a crucial factor which allows the system to be scalable (tens to hundreds WSCs) and easily reconfigured when there is a change in user requirement and/or the environment. The proposed approach also avoids the need for calibration boards (see [98]), allowing the approach to work with low resolution, wide FoV cameras.

5.3 Theory

5.3.1 Ground Plane Calibration and Back Projection

First of all, it is assumed that the surveillance networks are used to monitor activities that take place on a common plane (or approximately planar) [27, 28, 34, 39, 62, 63, 80, 108, 112] — e.g. people walking on a particular level of a building, cars entering and leaving a car park etc. In these cases, the perspective effect introduced by the projection of 3D world points onto the image plane can be safely reduced to a mapping between the ground plane and the image plane of the camera, provided that perspective lens are used, i.e. not fish eye lens. This mapping is referred to as the homography of the ground plane and is illustrated in Figure 5.1. More information related to homographies of planar scenes can be found in Section 3.2.3.

The key requirement for computing the homography is obtaining a number of

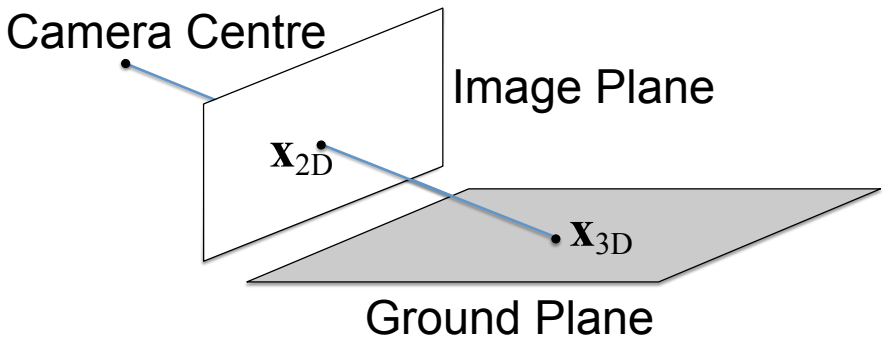


Figure 5.1: Perspective projection of a point on the ground plane to the image plane of the camera.

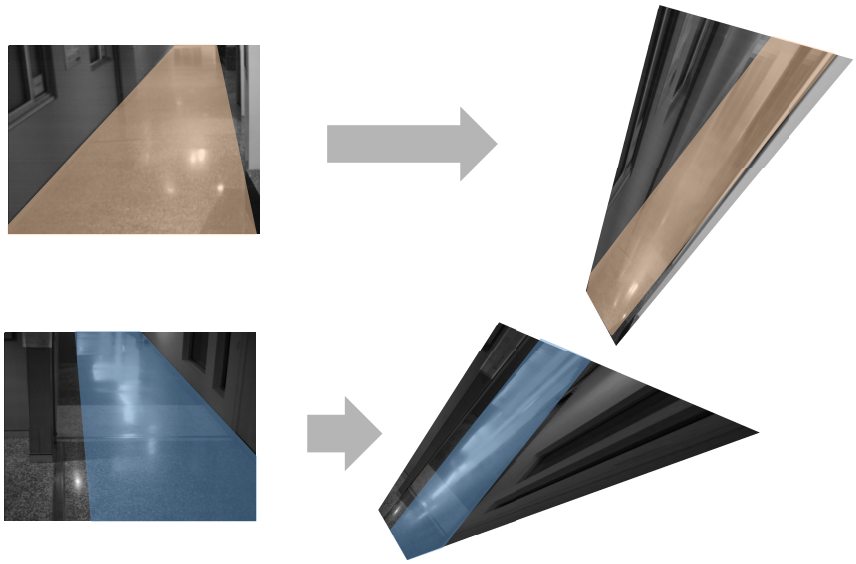


Figure 5.2: Images of two camera views transformed on to the same plane. In the resultant image, the floor areas are correctly mapped onto different parts of the same corridor, which is viewed from the top. Also notice that the walls in the resultant image are completely meaningless since they are not on the ground plane.

corresponding image-plane and ground-plane points. The ground-plane point coordinates are provided by a moving robot. As the robot moves it broadcasts its location to nearby cameras. The cameras are capable of performing motion

segmentation and blob analysis to determine the location of the robot in the image plane. Once a number of point correspondences are recorded, the homography is computed using a standard Direct Linear Transform as presented in Section 3.2.3.

The homography is a non-singular matrix which allows the world coordinate of an object of interest to be computed through back-projection,

$$\mathbf{x}_{3D} = \mathbf{H}^{-1}\mathbf{x}_{2D}. \tag{5.1}$$

For a tall object, such as a person, the inverse homography must be applied to the image point which lies on the ground plane, that is, the image coordinate of their foot and not their head. The greatest advantage of using such a system is that not only can the homography of individual cameras be computed automatically, the homographies also project points from different image planes onto the same ground plane, thus allowing objects to be localised seamlessly across different camera views.

5.3.2 Uncertainty Analysis

It is important to estimate the reliability of the system. The computed robot locations on the ground plane and the automatic detections of the robot in the images inevitably contain errors. The relationship between each individual component and their contributions to the back-projection uncertainty is captured in Figure 5.3.

Sankaranarayanan et al. [103] described a method for computing the effect of transforming a Gaussian-distributed random variable by a homography, however the implicit assumption of their work is that the homography computed is accurate, which is acceptable if the calibration points are selected manually. In the

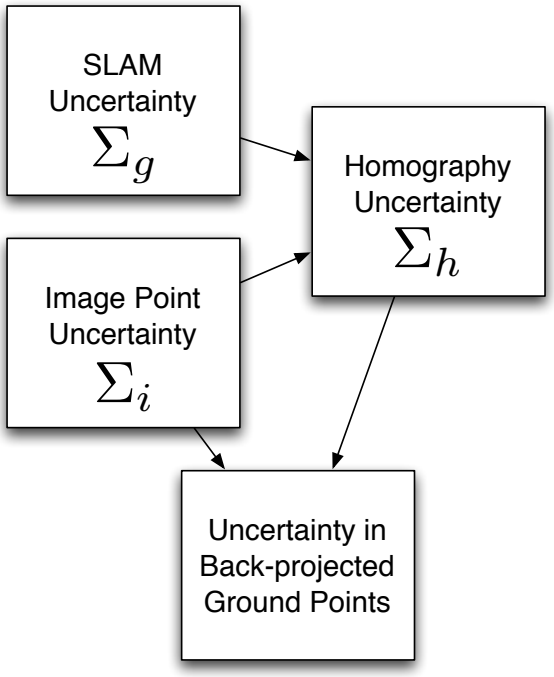


Figure 5.3: The causes of uncertainty associated with back-projected object locations.

proposed system, calibration points cannot be accurately determined since the centroid of the robot in the image plane does not precisely describe the location of the robot. In addition, the robot location reported by the SLAM algorithm is associated with an error. Both sources of errors indicate that the homographies computed are not perfect therefore the first step is to compute the uncertainties in the homographies. To deal with this, a matrix perturbation theory [44] based approach [1] is used.

As described in Section 3.1.2, the homography can be computed using the Direct Linear Transformation as $\mathbf{Ah} = \mathbf{0}$, where $\mathbf{h}$ is the 3×3 homography expressed in a vector format. The $\mathbf{A}$ matrix is a $2n \times 9$ coefficient matrix, where n is the number of correspondence. Each pair of image-ground point correspondence contributes to two rows of $\mathbf{A}$ and for $n \geq 4$, $\mathbf{h}$ can be solved using techniques

such as singular value decomposition (SVD).

Assuming that both image points $[x_i, y_i]^\top$ and ground plane points $[X_i, Y_i]^\top$ are associated with homogeneous isotropic Gaussian noise in the form of covariance matrices: for the image points $\mathbf{\Sigma}_i = \sigma^2\mathbf{I}$ and for the ground points $\mathbf{\Sigma}_g = \Sigma^2\mathbf{I}$, then the 9×9 covariance of $\mathbf{h}$ is,

$$\mathbf{\Sigma}_h = \mathbf{JSJ}, \tag{5.2}$$

where $\mathbf{J} = -\sum_{k=2}^{9} \frac{\mathbf{u}_k \mathbf{u}_k^\top}{\lambda_k}$, with $\mathbf{u}_k$ the k^{th} eigenvector of the matrix $\mathbf{A}^\top\mathbf{A}$ and λ_k the corresponding eigenvalue. $\mathbf{S}$ is the 9×9 matrix:

$$\mathbf{S} = \sum_{i=1}^{9} \left(\mathbf{a}_{2i-1}^\top \mathbf{a}_{2i-1} f_i^o + \mathbf{a}_{2i}^\top \mathbf{a}_{2i} f_i^e + \mathbf{a}_{2i-1}^\top \mathbf{a}_{2i} f_i^{oe} + \mathbf{a}_{2i}^\top \mathbf{a}_{2i-1} f_i^{oe} \right)$$

where $\mathbf{a}_i$ is the i^{th} row vector of matrix $\mathbf{A}$ and

$$\begin{aligned} f_i^o = \quad & \sigma^2[h_1^2 + h_2^2 - 2X_i(h_1h_7 + h_2h_8)] + 2\Sigma^2(x_ih_7h_9 + x_iy_ih_7h_8 + y_ih_8h_9) + \\ & (\sigma^2X_i^2 + x_i^2\Sigma^2)h_7^2 + (\sigma^2X_i^2 + y_i^2\Sigma^2)h_8^2 + \Sigma^2h_9^2, \end{aligned}$$

$$\begin{aligned} f_i^e = \quad & \sigma^2[h_4^2 + h_5^2 - 2Y_i(h_4h_7 + h_5h_8)] + 2\Sigma^2(x_ih_7h_9 + x_iy_ih_7h_8 + y_ih_8h_9) + \\ & (\sigma^2Y_i^2 + x_i^2\Sigma^2)h_7^2 + (\sigma^2Y_i^2 + y_i^2\Sigma^2)h_8^2 + \Sigma^2h_9^2, \end{aligned}$$

$$f_i^{oe} = \sigma^2[(h_1 - X_ih_7)(h_4 - Y_ih_7) + (h_2 - X_ih_8)(h_5 - Y_ih_8)].$$

The uncertainty analysis needs to go one step further: Given the covariance matrix of a homography $\mathbf{\Lambda}_h$ and the image plane location of an object of interest $\mathbf{x}$, with an error covariance $\mathbf{\Sigma} = \sigma^2\mathbf{I}$, what is the uncertainty of the back-projected

$\mathbf{X}$. The back-projected point $\mathbf{X}$ can be computed by an alternative form of $\mathbf{Ah} = \mathbf{0}$ as,

$$\mathbf{X} = \mathbf{Bh}, \tag{5.3}$$

where $\mathbf{h}$ is the vectorised form of the homography matrix $\mathbf{H}$ and $\mathbf{B}$ is the image plane point $\mathbf{x}$ expressed as,

$$\mathbf{B} = \begin{bmatrix} \mathbf{x}^\top & \mathbf{0}^\top & \mathbf{0}^\top \\ \mathbf{0}^\top & \mathbf{x}^\top & \mathbf{0}^\top \\ \mathbf{0}^\top & \mathbf{0}^\top & \mathbf{x}^\top \end{bmatrix}.$$

Then the uncertainty in $\mathbf{X}$ can be computed as,

$$\mathbf{\Lambda}_X = \left[\begin{array}{c|c} \mathbf{B} & \mathbf{H} \end{array}\right] \left[\begin{array}{c|c} \mathbf{\Lambda}_h & \mathbf{0} \\ \mathbf{0} & \mathbf{\Lambda}_x \end{array}\right] \begin{bmatrix} \mathbf{B}^\top \\ \mathbf{H}^\top \end{bmatrix}. \tag{5.4}$$

5.3.3 Sparse Object Tracking

Once the camera network is calibrated in terms of the homographies of the ground plane, the network can be used to localise objects of interest that move within the region covered. The raw output from each camera, computed by Equation (5.1), is the object's location at discrete times which contains error. A particle filtering process is used to link these location estimates to better compute the object's true locations. The particle filtering process uses a Monte Carlo based multi-hypothesis estimation algorithm [113], which can be derived from recursive Bayesian estimation, as a three stage process consisting of:

1. *Prediction:* Predict the position of the object using the received message of target's current location, estimated speed and heading and previous posi-

tion.

2. *Correction:* Ensure that the position of the object is valid according to the map floorplan.

3. *Resample:* Select new or valid position(s) to represent the current location of the object.

Recursive Bayesian Estimation can be described by the predicted probability term: $p(\tilde{\mathbf{x}}_k) = p(\mathbf{x}_k|\mathbf{z}_0, \ldots, \mathbf{z}_{k-1})$ as seen in Equation (5.5),

$$p(\tilde{\mathbf{x}}_k) = \int p(\mathbf{x}_k|\mathbf{x}_{k-1})p(\mathbf{x}_{k-1}|\mathbf{z}_0, \ldots, \mathbf{z}_{k-1})\mathrm{d}\mathbf{x}_{k-1} \tag{5.5}$$

The term $p(\mathbf{x}_k|\mathbf{x}_{k-1})$ is a predicted probability of the object's real-time position based on the camera nodes' object locator. The probability distribution $p(\mathbf{x})$ is represented by a number of N weighted samples $\mathbf{x}^{[i]}, i = 1..N$, with weight factors $w^{[i]}$ as seen in Equation (5.6).

$$p(\mathbf{x}) \approx \sum_i w^{[i]}\delta(\mathbf{x}^{[i]} - \mathbf{x}). \tag{5.6}$$

The discrete samples represent *particles* or the possible positions of the object. The weight factor $w^{[i]}$ of each *particle* determined the validity of the predicted position.

Prediction Stage

The prediction stage calculated a new set of *particle* positions based on object locator updates from the camera node and using the previous position and speed

of the object. If there are no updates from the camera node, the particle filtering process will use the object's last known speed and heading value to predict a new position.

$$\begin{pmatrix} p_x \\ p_y \end{pmatrix}_k^{[i]} = \begin{pmatrix} p_x \\ p_y \end{pmatrix}_{k-1}^{[i]} + \hat{l}_k^{[i]} \cdot \begin{pmatrix} \sin(\hat{\phi}_k^{[i]}) \\ \cos(\hat{\phi}_k^{[i]}) \end{pmatrix} \tag{5.7}$$

where i is the particle index and k is the time.

Correction and Resampling Stages

The correction stage determines the validity of the *particle* positions by detecting if any physical barriers are crossed (i.e. walls, etc). The weight factors in Equation (5.6) are updated by the correction stage, using Equation (5.8).

$$w^{[i]} = \tilde{w}^{[i]} \cdot p(\mathbf{z}_k|\tilde{x}_k^{[i]}), \quad \sum_i w^{[i]} = 1. \tag{5.8}$$

where $p(\mathbf{z}_k|\tilde{x}_k^{[i]})$ is the probability of observing the current measurement (which is sent by the smart cameras) given a particular particle.

The weight values depend on the following conditions which are determined if the *particle* has crossed a barrier on the floor-plan.

$$\tilde{w}_k^{[i]} = \begin{cases} w_{k-1}^{[i]} & \text{no wall crossed} \\ 0 & \text{wall crossed} \end{cases} \tag{5.9}$$

The floor-plan map of the tracking area is used to check the validity of the object's estimated position. The estimated position is considered invalid if the object has

to move through a barrier. When this occurs, the position is re-estimated until it is valid. When a barrier is detected, the weight factor is updated according to Equation (5.8).

Once the weight factor values have been determined, new *particles* are created by resampling the current set of *particles* according to each weight factor.

5.4 System Framework

5.4.1 Overview

The core problem this chapter addresses is the calibration of non-overlapping DWSCs installed in a man-made environment. Locations in the environment can be viewed by zero (un-overlapped FoV), one or more cameras, whose positions and orientations are unknown. The set-up of the network is shown in Figure 5.4.

The proposed system has two phases of operations: the camera network is first calibrated and then used to localise objects of interest. Instead of explicitly determining the positions of the cameras, mappings from each camera's image plane to a global coordinate frame are created. During calibration, a robot with the capability of computing its own locations on the ground plane is employed. When it moves into a camera's FoV, the corresponding image plane coordinates is extracted and used to establish the mapping between the ground plane and the image plane of the camera.

Once the calibration has been determined the network can be used to track objects. Objects are detected and the image plane coordinates are mapped to an estimate of the object's world coordinate, allowing the object to be tracked through

a disjoint wireless smart camera network. The framework of the calibration and localisation system is presented in this section and the specific implementation using off-the-shelf hardware will be described in Section 5.5.

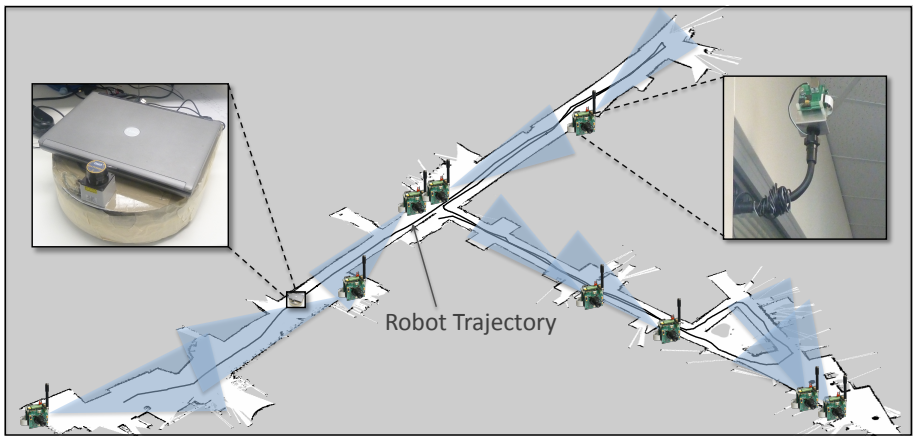


Figure 5.4: Illustration of layout of wireless smart cameras and intersection of camera FoV with robot trajectory. The shaded regions represent the FoV of each camera.

5.4.2 System Components

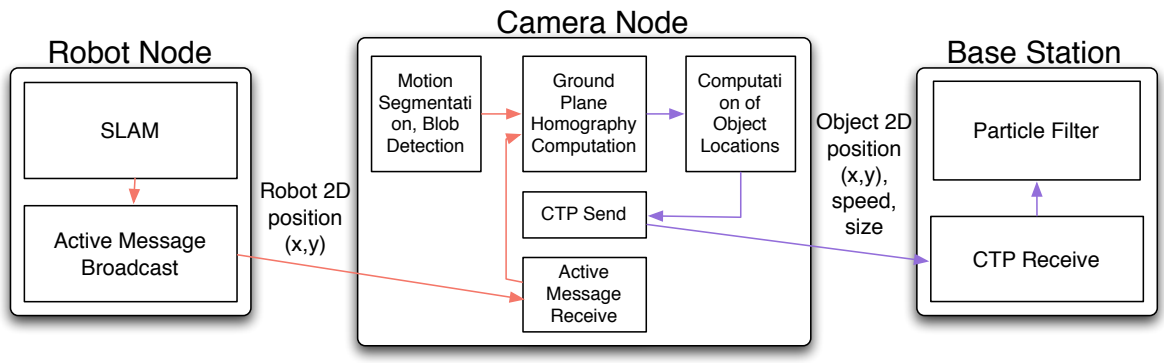


Figure 5.5: Illustration of key software components of system.

Key components of the proposed system are: robots, wireless smart cameras and

a network base-station as shown in Figure 5.5. Wireless smart cameras are responsible for detecting and tracking objects within their FoV and communicating with the robot in order to determine the mapping between image and ground plane. They also run a collection tree protocol (CTP) stack [43] in order to provide reliable wireless multi-hop transmission of localisation information to the base station.

Wireless Smart Camera Network

The smart cameras are required to include a camera, a processing unit with sufficient memory for image processing tasks and a wireless radio node. Images are captured at low resolution, typically 320×240 or even 160×120 to allow real-time performance.

Each smart camera contains five software components (Figure 5.5). Sitting at the bottom is the active message layer which communicates with the mobile robot through a one-hop broadcasting scheme. The second layer is the collection tree protocol (CTP) which provides reliable multi-hop transmission to the base station. The other three components support object localisation. The motion segmentation module detects moving objects in the image and computes their centroids/foot locations. The difference between the centroid and the robot's true location on the camera's image plane is not neglected although the robot's physical dimension is relatively small (circular shape with 40cm diameter) comparing to the robot-camera distance (around 5m to 10m). This difference is accounted by the assumption that the measured image plane locations are associated with homogeneous isotropic Gaussian noise of covariance $\mathbf{\Sigma}_i = \sigma^2\mathbf{I}$. The impact of this error has been discussed in Section 5.3.2. During the calibration phase, the centroid of the robot and the received real-world location of the robot is used by the homography computation component to estimate the homography

induced by the ground plane. During the localisation phase the inverse of the homography is used to map the image-plane locations (centroids for tracking a robot, or foot locations for tracking human) to the ground plane which is then sent to the base station by the CTP component.

Mobile Robot

The mobile robot uses laser sensors and odometry to update a map of the world and to estimates the robot's position, with respect to a global coordinate frame, using a Simultaneous Localisation and Mapping (SLAM) algorithm [47]. The position estimates are broadcast to the cameras to estimate the ground-plane homography. The local region of the map can also be broadcast to the cameras to indicate walls which are used later to constrain the target motion.

Base Station

The base station uses a collection tree protocol or similar protocol such as 6LowPan to receive object location messages sent by the cameras. Each object location message contains the x and y positions, speed (in meters/second) and size of the object (in terms of the number of pixels). The base station also receives the local map regions broadcast by the robot during the calibration phase.

5.5 Evaluation

5.5.1 Experimental Setup

The mobile robot used is an *iRobot Create* (http://www.irobot.com/Create) research platform equipped with an Hokuyo scanning laser range finder, a laptop computer running Linux and wireless sensor node connected via a USB serial port (Figure 5.4). The SLAM software, from ROS (Robot Operating System from http://www.ros.org), runs on the laptop and uses the sensor data to continuously update a map of the environment and the robot's position. The robot node continually broadcasts its position estimate via the attached wireless sensor node.

A network of nine low-power wireless smart cameras [89, 123] are deployed inside a building at an average spacing of approximately $10m$ (Figure 5.4). The network is first calibrated by the autonomously driven *iRobot Create*. The robot was programmed to visit the approximated FoV of each camera, which was based on the knowledge gained during installation of cameras. For each coarsely known FoV, the robot visits and broadcasts its positions at 16 locations, ensuring a high likelihood of at least 10 of them being within the actual FoV of each camera. Although a minimum of 4 pairs of correspondences are needed, 10 pairs were used to improve the accuracy of the solution. In order to evaluate the localisation performance of the wireless smart camera network, the same robot is used as the tracking target. Although any reasonable size object can be used, the SLAM data of the moving robot (which during localisation phase is not relayed to the cameras or base-station) serves as ground truth to evaluate the accuracy of tracking.

5.5.2 Results and Discussion

Three analysis were conducted to evaluate the proposed system: the uncertainty of the computed calibration for each camera, the tracking accuracy of the calibrated network, and the importance of each camera to the performance of the network.

Back Projection Uncertainty

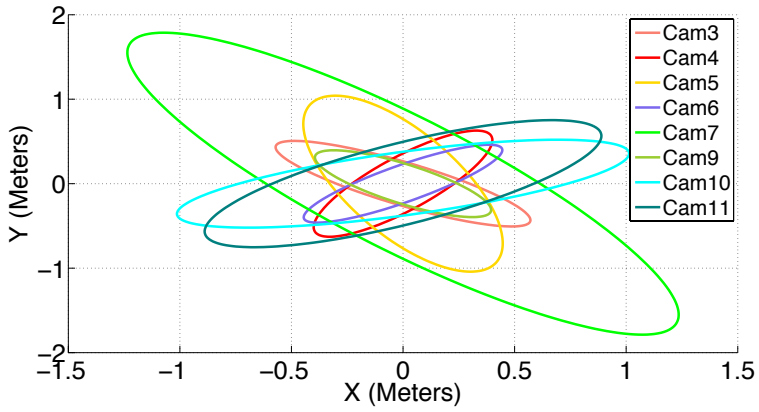


Figure 5.6: Average uncertainty regions for all cameras.

The reliability of the back-projected target locations is degraded by uncertainties in both the calibration and tracking stages. To analyse the reliability, the uncertainties in the homography is computed first and then combined with the uncertainties in target locations on the image planes to find the uncertainty regions of the tracked object on the ground plane. This concept is summarised in Figure 5.3. For each camera the uncertainty covariance matrix is averaged across all tracked object locations, and is presented in Figure 5.6 as ellipses of 90% confidence level. The same result is drawn in Figure 5.8 for each tracked location of the moving object from all cameras. It can be seen that most SLAM output

falls within the uncertainty regions, indicating that the uncertainty calculated is accurate.

Out of the 8 cameras (as camera 12 failed), camera 7 has the largest uncertainty due to the fact that the homography was computed using point correspondences that were mostly located in the left half of the image, as shown in Figure 5.7. Better path planning would ensure that points on the right half of the image are also used in calibration to improve the result. Knowing the uncertainty regions enables camera selection for improving multi-camera tracking accuracy or for the purpose of energy saving – topics of future investigation. With two-way communication between robot and camera during calibration, it would be possible to obtain better distributed calibration points.

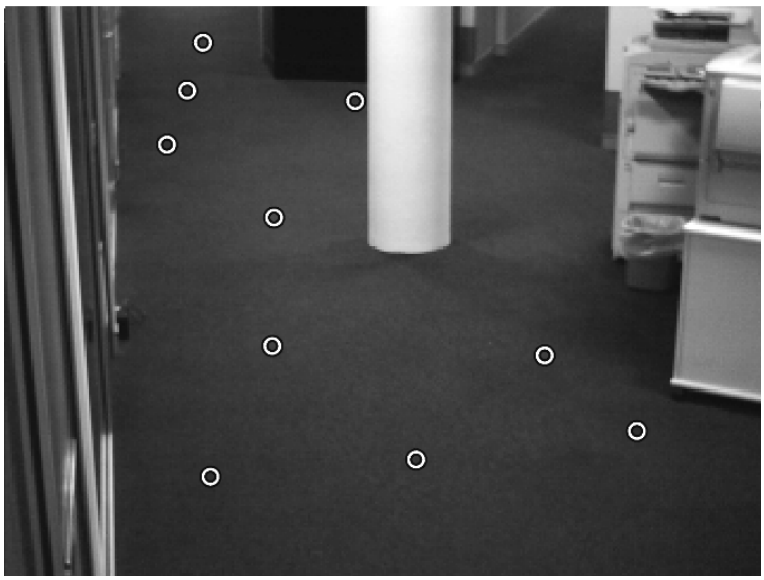


Figure 5.7: Image points used to calibrate camera 7. There is a high concentration of points on the upper-left corner of the image, and the upper-right corridor is not covered by any calibration points. This is the main cause of inaccuracy when the robot travels in the upper-right corridor.

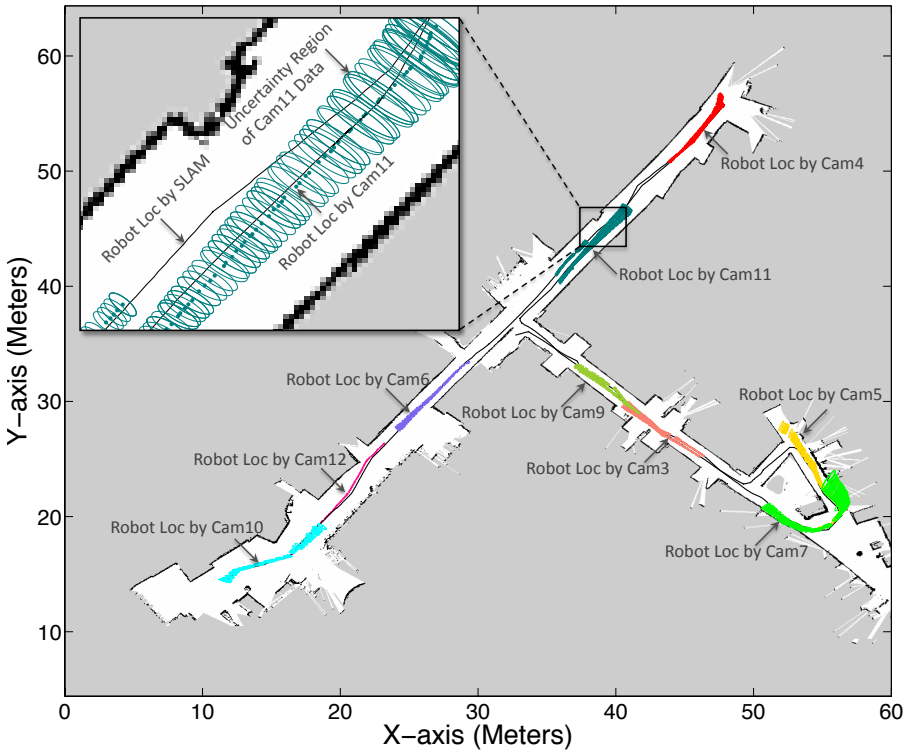


Figure 5.8: Tracking results with uncertainty regions. Uncertainty regions for camera 12 is not available due to hardware failure.

Tracking Accuracy

Since it is impractical to obtain ground-truth data about the position of the tracked target in a large deployment area, the robot is used as the target so that the trajectory of the robot reported by the camera network can be compared with the robot's own estimate of position determined by its onboard SLAM system.

To evaluate the accuracy of the system, three different metrics are computed: the cumulative distribution function of the error of 1) the raw data, 2) the unprocessed object locations sent from each wireless smart camera, and 3) the continuous data which is the output of the particle filter on the base station. Error is defined as the distance between a measured/estimated location vector and the SLAM

location vector $[x, y]$ at the same time instance. The results are summarised in Figure 5.9(a) and Figure 5.9(b). It can be seen that the over 80% of the unfiltered data points are within 0.4m of the ground truth locations, while 80% of the filtered data points are associated with an error of less than 2m. Note that the particle filter attempts to recover the target trajectory when it is unobservable by making use of the current speed, heading and the map. The error is primarily caused by the fact that the robot changed its heading a few times, which the particle filter assumed to be constant.

Camera Importance Analysis

To assess the importance of individual cameras on overall tracking performance, the particle filter response is calculated when a particular camera is removed from the network. The results are shown in Figure 5.10. It is interesting that the uncertainty of a camera (see Section 5.5.2 and Figure 5.6) is not directly related to its effectiveness from a target localisation perspective. For example, camera 9 has the smallest uncertainty ellipse in Figure 5.6, but its impact on the network is limited, since removing it does not greatly reduce the localisation performance. Cameras 4 and 10 are at the end of the corridor, where the robot turns around, thus removing them will have significant impact. This suggests that cameras that are at decision points such as dead-ends, doorways and corners are more important than others. It is also observed that when two cameras overlap, removing one of them will not produce a large error, example pairs are cameras 3 and 9, and cameras 5 and 7. Removing camera 12 gives the largest error due to the violation of the assumption that the robot moves in a linear fashion in the unobservable regions.

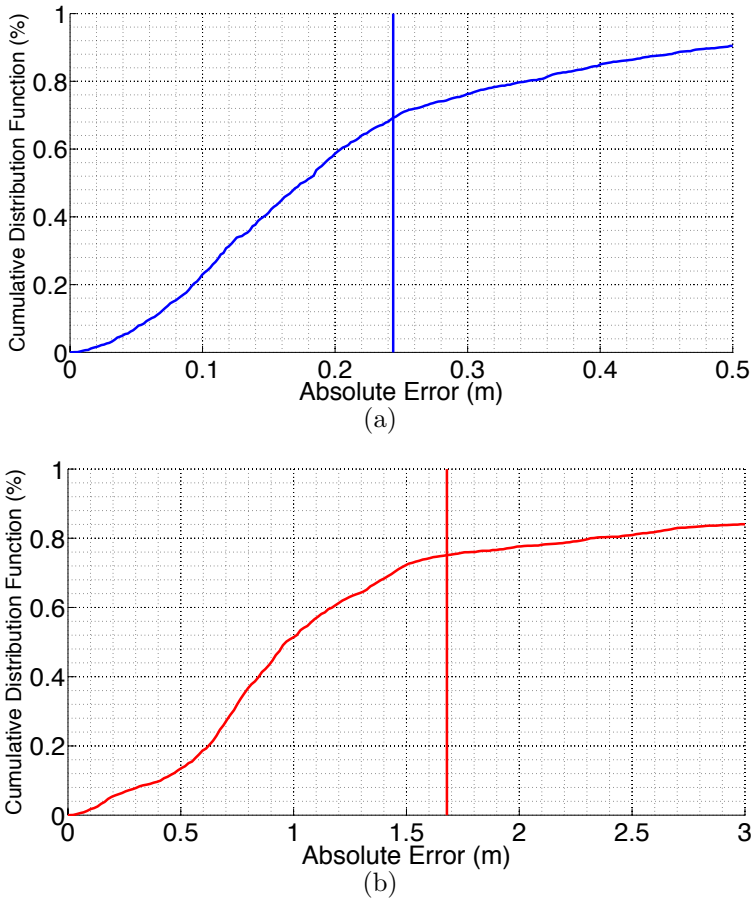


Figure 5.9: (a) Error of the raw data. Raw data does not cover the unobserved area. (b) Error of the tracking output. Tracking output is continuous since the object location is propagated based on last known speed, heading and the map in the unobservable regions. Vertical lines in both (a) and (b) indicate the average error.

5.6 Chapter Summary

This chapter has described an innovative approach to camera network calibration and object localisation using a network of non-overlapping wireless smart cameras and a mobile robot.

The robot provides global position data to determine each camera's local image

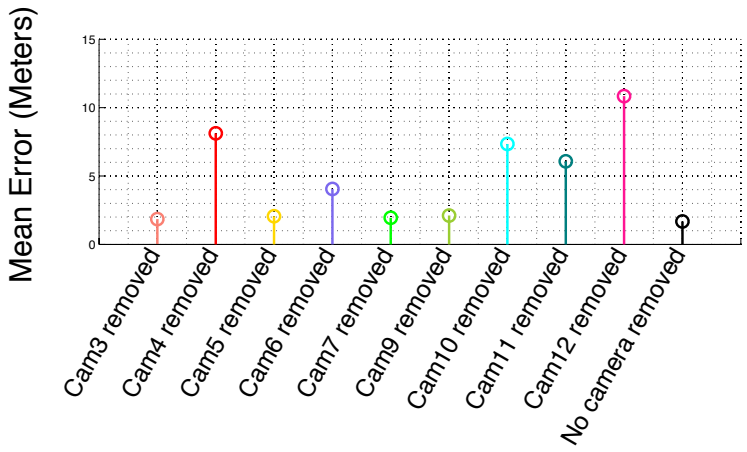


Figure 5.10: Error of the particle filter output when a camera is removed.

to ground plane mapping, as well as a global occupancy grid to support a map-based tracking algorithm. Experiments have been conducted using a twelve-node indoor wireless smart camera network, and the results indicate high levels of object localisation accuracy (80% of the localisation errors are less than 0.4m) once ground-plane homographies have been determined during the robot-aided camera calibration phase. Furthermore, the uncertainty of each camera as well as its individual effectiveness has also been shown. Effectiveness is useful in allowing the user to select the best camera among overlapping cameras in real time and/or choose to reposition cameras for extended coverage.

The approach differs from traditional approaches in a number of ways. Importantly, the proposed approach does not require overlapping camera views which is crucial for large-scale environments. Compared to other calibration systems consisting of non-overlapping cameras, the proposed approach does not involve human input, does not require any calibration board, and is applicable to low resolution imagery and cameras with wide FoVs. The autonomy of the system is critical to high scalability (tens to hundreds WSCs) and flexibility (easy to redeploy). Therefore, objective 2 listed in Section 1.2.2 has been successfully

achieved.

Chapter 6

Object Association for Tracking in Non-overlapping Wireless Smart Cameras

6.1 Introduction

As low-power, pervasive computing becomes an increasingly viable paradigm for many applications, topics such as visual object tracking are finding a renewed interest in the computer-science research community. The notion of object tracking is to continuously estimate the positions and attributes of an object of interest (e.g. people) within a defined observation space. When applied to distributed wireless smart camera networks, the problem becomes one of forming a global observation space from the local observation spaces occurring at each camera. In this way the local state of objects within each node observation space can be translated into a global state space.

Over the past two decades, a significant amount of research around multi-camera tracking systems has been based on centralised systems (e.g. [126]). In these designs, significant computational resources are assumed to be present in the centralised hub of such systems. Energy consumption, communication bandwidth, and reliability between cameras are therefore not dominant system constraints. As such, most of the techniques developed do not map well when applied within the constraints of low-power, wireless distributed systems where computational resources and communication links are clearly constrained. To assist in this situation recent work has addressed some of the challenges that exist in distributed tracking systems, e.g. [55, 66]. However, most of the techniques proposed still assume significant resources are available at each of the nodes.

This chapter addresses what is believed to be some of the key issues which are yet to be fully explored in distributed non-overlapping multi-camera tracking systems. By taking into account the dominant constraints which are introduced by low-power, pervasive camera smart networks, a new set of research questions are introduced for this class of systems. In particular, the problem of *target handover*, a well studied topic within the computer vision community [16, 42, 58, 61], is considered within a disjoint camera network. The main contribution of this chapter is the proposal of a master/slave mechanism [95] based two-stage network level protocol to handover. The first stage considers the forwarding of the object descriptions to the neighbouring cameras, based on a coarsely known or learnt topology of the network. The second stage employs a probabilistic matching approach to determine the most likely observation that matches to an object entering the FoV from a number of received observations. This approach improves the accuracy of the existing master/slave mechanism and is suitable for distributed platforms of limited resources.

6.2 Related Work

The main core of this work is inspired by that of Javed et al. [58, 59] who presented an approach for establishing object correspondence across cameras with non-overlapping views. Their method exploits the redundancy in paths that objects such as people and cars tend to follow. The system has a learning phase where the camera topology and path probabilities are estimated using a Kernel Density Estimator over a small number of manually labelled trajectories and updated with changing trajectory patterns. The authors in [57] presented an approach to track cars on a highway, modelling the colour and transition times as Gaussian distributions. Their method requires calibrated cameras and relies on the fact that cars move in the same direction. The methods developed in both [58] and [57] are centralised approaches however, and therefore do not work well on distributed systems in which nodes do not have complete knowledge of the overall system.

In contrast to [58] and [57], Ellis et al. [32] did not require apriori correspondences or a training phase, instead they used the observed motion over time to establish reappearance periods. Many other probabilistic or statistical methods do not make use of the assumption that objects tend to follow some trajectories. These include the work of Tieu et al. [114] who focused on measuring statistical dependence between observations in different cameras. The nature of this dependence is characterised by the distribution of observation transformations between cameras, such as departure to arrival transition times, and colour appearance.

More recently, researchers have begun to consider the multi-camera tracking problem in a distributed fashion in smart camera nodes (wired smart cameras) where energy is not a real concern. Quaritsch et al. [95] described a distributed embedded system for tracking multiple objects. Although light-weight, the master/slave handover mechanism associates migration regions with one or more subsequent

cameras on the path. This relies on the node level tracking algorithm to be able to precisely split the image into multiple migration regions. Computing migration regions may not be feasible on low-power wireless camera nodes where computation capabilities are limited for two reasons: First, the light weight tracking implemented on each node may not be able to precisely detect the exit and entrance locations. Second, if the nodes are to be duty cycled, or triggered in order to reduce energy consumption, migration regions cannot be reliably detected. Furthermore, the slaves match objects based on purely the appearance (colour) of the objects. Spatio-temporal relationships between disjoint observations of the same object are completely ignored. The master/slave handover mechanism was adopted by Hoffmann et al. [55] who introduced a set of PTZ camera management protocols to allow pre-selected targets to be tracked seamlessly across the network.

In [17], Camp et al. proposed to use a Hidden Markov Model (HMM) to model the state changes of objects as they move between different cameras. Kim et al. [66] employed Markov chain Monte Carlo to split the entire observation set into a number of subsets, each of which corresponds to the track of an object. The latter two approaches may require extensive computation and may not be feasible for low-power devices.

6.3 Distributed Multi-camera Tracking

This section formally describes the system framework, including the formulation of the problem of distributed object tracking and the resultant network level protocols which form the underlying primitives of the system.

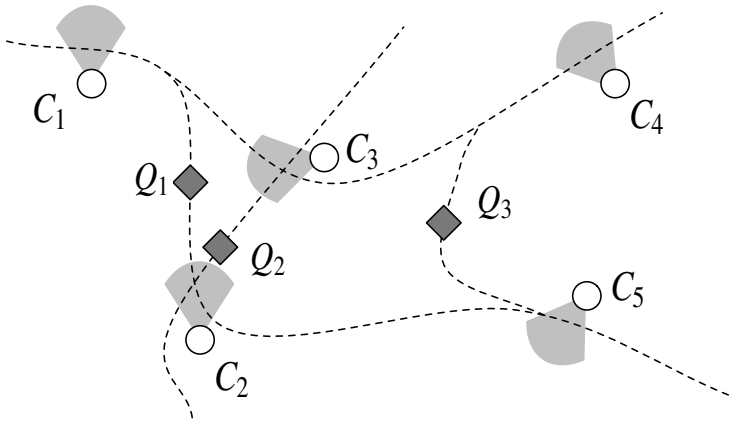


Figure 6.1: Outline of multiple camera topology, with multiple routes between cameras. $C_1 ... C_5$ are cameras and Q_1, Q_2, Q_3 are three different objects.

6.3.1 Problem Overview

The system is designed around the goal of tracking multiple objects across multiple cameras where a global knowledge of the space in which objects can move is not directly available. Instead, knowledge about the global space must be derived from the local observations which are captured by each camera. It is assumed that objects move along a finite number of paths between cameras where cameras are orientated in such a fashion as to ensure objects will, at some point, pass through their FoV. This system design is illustrated in Figure 6.1. Many of the current tracking systems have been developed with infrastructure based networks in mind. Using a centralised approach, a global knowledge of the network is therefore maintained at all times. The energy-constrained wireless smart cameras considered in this project necessitate a distributed approach to the problem where the assumption of complete global knowledge is no longer valid.

Deriving global knowledge of the entire network in an energy-constrained wireless smart camera network is not a trivial task and invites many interesting research

challenges. While it may be possible for all nodes to constantly flood the network with status information, this is clearly an unreliable solution as it will introduce large delays in the network, compromising tracking performance, and limiting the scalability of the network.

However, it can be argued that a global knowledge of the network at every camera is not necessary and all that is required is knowledge about a camera's nearest neighbours on a potential object path. With this observation in mind, it is introduced in this section a predictive-based tracking strategy which enables the system to exploit knowledge of the likely paths of objects to improve performance and reduce communication overheads. If two cameras are situated on the same path, there is a high probability that the same object will be observed in both cameras in a sequential manner. It is therefore one of the goals of the proposed predictive tracking strategy to determine this behaviour over time. This leads to the following advantages over non-predictive distributed systems:

1. Improved tracking performance (object miss, ID loss and ID switch).
2. Reduced communication overhead.

6.3.2 Problem Formulation

A network of N cameras is defined as $\mathbf{C} = \{C_1, C_2, C_3, ..., C_N\}$. The cameras are placed in such a way that each of them monitors a section of a path or an intersection of paths along which one or more objects can move. The cameras may assume a coarse network topology so that the system can quickly reach steady state, otherwise, the topology can be learnt over a longer period of time. In this setting, non-overlapping cameras are assumed as object hand over in an overlapping camera network can take the advantages of the relative geometry of

the camera and thus is not as difficult (See Section 3.2.3 for more information). Given that the cameras wirelessly communicate with one another, each has a set of *radio* neighbours. These neighbours are restricted to be cameras that are one-hop away in terms of radio range. For each camera C_i, this set is denoted by $\mathbf{R}_i$, where $\mathbf{R}_i \subset \mathbf{C}$.

If assuming that a set of physical objects $\mathbf{Q} = \{Q_1, Q_2, \ldots, Q_M\}$ are passing through the observation space of the camera network, then the global set of observations of the camera network can be defined as:

$$\mathbf{O} = \{O_i^\tau | i \in \{1, 2, \ldots, N\}, \tau \in \{a, b, \ldots, M\} \\ \text{where} \, \forall O_i^\tau, O_j^\delta, \text{if} \, i \neq j \, \text{then} \, \tau \neq \delta\} \quad (6.1)$$

The notation O_i^τ is the network observation τ obtained at camera C_i. The alphabetical order of specific values of τ (i.e. a, b, c etc.) explicitly represents the order in which the observations were captured. The actual physical objects represented by the observations O_i^a, O_j^b, O_k^c are denoted as Q_i^a, Q_j^b, Q_k^c respectively. It is possible that $Q_i^a = Q_j^b$ if O_i^a and O_j^b are both observations of the same physical object. The problem of tracking can therefore be formulated as dividing the observation set $\mathbf{O}$ into a number of subsets, with elements of each subset representing the same physical object. To achieve this in a distributed environment, an effective approach is introduced which consists of a predictive forwarding strategy and a probabilistic matching method.

6.3.3 Predictive Forwarding

Consider the case in which an object leaves the FoV of camera C_i. In the master/slave mechanism, the descriptive information is relayed to a subsequent camera depending on the migration region, from which the object leaves. However,

this region may not be available. In the case of a duty cycled schedule, for example, the camera may be in sleep mode at the time the object leaves the FoV of the camera. In other cases it might only be possible to use more simple (and less effective) local tracking algorithms which also lead to inaccurate information about the migration region.

Using a centralised approach the corresponding observation O_i^τ would be maintained by the server. However, with a distributed approach the descriptive information must be relayed to another camera in order to track the object. To help address the problem, a predictive forwarding strategy is introduced which forwards an objects descriptive information based on the object's *last known location* instead of the migration regions. Ideally an object's last known location is always available as the object is tracked. Two sets are defined for a camera C_i: the *preceding* set, and the *succeeding* set. The former set, denoted by $\mathbf{L}_i$ is the set of cameras adjacent to C_i *from which* the object might have come. The latter set, denoted by $\mathbf{K}_i$, is the set of cameras adjacent to C_i *to which* the object may go. Two cameras are considered *adjacent* if it is possible to move from one camera to the other without passing through the FoV of a third camera. In the case where the routes between the cameras are bidirectional then $\mathbf{L}_i = \mathbf{K}_i$.

As the predictable nature of paths objects normally take is explicitly exploited, there exists a number of different probability distributions for cameras in set $\mathbf{K}_i$. For example, consider objects Q_1 and Q_2 in Figure 6.1. If it is assumed that they are both moving towards camera C_2, then clearly the probability distributions of their next destination will be significantly different at the time they leave camera C_2. This is marked by the fact that they have come from different source cameras (C_1 and C_3, respectively). Denoting the source camera as C_s, the objects'

observation is forwarded to the set of cameras:

$$\mathbf{F}_i^s = \{C_j | C_j \in \mathbf{K}_i, P(C_j|C_s) > T\} \tag{6.2}$$

where $P(C_j|C_s)$ is the probability of an object moving to camera C_j given that it has come from the source camera C_s and T is a predefined threshold quantifying the minimum transition probability associated with an succeeding camera. $\mathbf{F}_i^s$ can be therefore thought of as a filtered version of the set $\mathbf{K}_i$ given the extra information about the source of the object. If no source information is available, the observations will be forwarded to all cameras in set K_i. Once the information has been transmitted, it is the destination camera's responsibility to determine whether the object has entered its FoV and to provide an acknowledgement to camera C_i upon a successful matching. The probability $P(C_j|C_s)$ is then re-estimated using a sliding window approach when this feedback is received.

6.3.4 Probabilistic Matching

As explained in Section 6.3.3, the current camera C_i will multi-cast the object descriptor to all members of the set $\mathbf{F}_i^s$. Any particular member, C_j, of this set will receive the object descriptor along with other possible descriptors from cameras in its own proceeding set $\mathbf{L}_j$. Upon subsequent detection of an object, C_j must determine which descriptor in its received list correctly matches the current object.

To assist in the explanation following notation is defined for clarity: $Q_i^\tau \rightleftharpoons Q_j^\delta$ which says that the objects are the same physical objects and that the object entered C_j after leaving C_i. Given an observation O_j^δ in camera C_j, a hypothesis set $\mathbf{H}_j^\delta$ is defined to be the set of possible matches to O_j^δ. This set of observations is generated over the subset of $\mathbf{L}_j$ that detected an object and transmitted to

camera C_j before O_j^δ is determined. For any observation, O_i^τ, taken from the set $\mathbf{H}_j^\delta$, the probability that O_j^δ matches this is given by:

$$P(Q_i^\tau \rightrightarrows Q_j^\delta | O_i^\tau, O_j^\delta) = \frac{P(O_i^\tau, O_j^\delta | Q_i^\tau \rightrightarrows Q_j^\delta) P(Q_i^\tau \rightrightarrows Q_j^\delta)}{P(O_i^\tau, O_j^\delta)} \tag{6.3}$$

The object handover problem seeks to maximise this probability over $\mathbf{H}_j^\delta$:

$$\begin{aligned} h &= \arg\max_{\forall O_i^\tau \in \mathbf{H}_j^\delta} \{P(Q_i^\tau \rightrightarrows Q_j^\delta | O_i^\tau, O_j^\delta)\} \\ &= \arg\max_{\forall O_i^\tau \in \mathbf{H}_j^\delta} \{\frac{P(O_i^\tau, O_j^\delta | Q_i^\tau \rightrightarrows Q_j^\delta) P(Q_i^\tau \rightrightarrows Q_j^\delta)}{P(O_i^\tau, O_j^\delta)}\} \end{aligned} \tag{6.4}$$

It is assumed that the observation pairs are uniformly distributed and thus $P(O_i^\tau, O_j^\delta)$ is a constant scaling factor and can be left out. The problem is then reduced to:

$$h = \arg\max_{\forall O_i^\tau \in \mathbf{H}_j^\delta} \{P(O_i^\tau, O_j^\delta | Q_i^\tau \rightrightarrows Q_j^\delta) P(Q_i^\tau \rightrightarrows Q_j^\delta)\} \tag{6.5}$$

For object movement between cameras, two properties are measured for matching: the change in appearance, ρ and the time of departure t_d and arrival t_a of the object. If assuming independence between appearance and time measurements, the above formula becomes:

$$\begin{aligned} h = \arg\max_{\forall O_i^\tau \in \mathbf{H}_j^\delta} \{ & P(O_i^\tau(\rho), O_j^\delta(\rho) | Q_i^\tau \rightrightarrows Q_j^\delta) \\ & P(O_i^\tau(t_d), O_j^\delta(t_a) | Q_i^\tau \rightrightarrows Q_j^\delta) \\ & P(Q_i^\tau \rightrightarrows Q_j^\delta)\} \end{aligned} \tag{6.6}$$

In order to maximise this probability, the three individual components need to be estimated:

The first component, $P(O_i^\tau(\rho), O_j^\delta(\rho)|Q_i^\tau \rightrightarrows Q_j^\delta)$, represents the probability of obtaining a particular similarity value between observations of the same object across the two different cameras given the distribution of the similarity values of observations of identical objects across the two different cameras. The similarity value between two descriptions can be computed using a *similarity* function $sim(O_j^a, O_i^\delta)$ which outputs a value from 0 to 1 depending on the confidence in a match. Examples of the similarity functions include major colour histogram [75] or the Bhattacharyya distance [58]. A Gaussian distribution for the similarity between the descriptions of the same object for each pair of neighbouring cameras can be obtained during a training phase. Thus the probability $P(O_i^\tau(\rho), O_j^\delta(\rho)|Q_i^\tau \rightrightarrows Q_j^\delta)$ for any two observations can be found using the trained Gaussian.

The second component, $P(O_i^\tau(t_d), O_j^\delta(t_a)|Q_i^\tau \rightrightarrows Q_j^\delta)$, represents the probability of the time transition $(t_d - t_a)$ given the distribution in the transition times between camera C_i and C_j. Similar to the appearance probability, a Gaussian distribution can be fitted to the arrival times of the same object between any particular pair of cameras and use this to derive the probability of any new observed arrival time. Although a single Gaussian distribution is used, other distributions may be more suitable in certain circumstances. Cai et al. [16] used a mixture of 3 Gaussian distributions to model this distribution whereas Javed et al. [58] did not assume any particular distribution and instead use a Parzen window to estimate it. Future work will consider different types of distributions in more detail to account for the difference in object speeds.

The final component, $P(Q_i^\tau \rightrightarrows Q_j^\delta)$, is the prior probability of arrival at C_j from camera C_i. This probability is calculated by determining the number of objects moving from camera C_i to C_j and dividing by the total number of objects arriving at camera C_j. To ensure the probability distributions correctly reflect the changes

in the environment or the object behaviour pattern, the Gaussian distributions are updated in real-time with the commonly-used sliding window approach:

$$\mu_t = ((T-1)/T)\mu_{t-1} + O_t/T \tag{6.7}$$

$$\sigma_t^2 = ((T-1)/T)\sigma_{t-1}^2 + (O_t - \mu_t)^2/T \tag{6.8}$$

where O_t is the newly calculated similarity score or the transition time and T is the size of the sliding window.

6.4 System Implementation

This section outlines the practical implementation of the distributed object tracking system that utilises the protocols described in Section 6.3. A two-layer structure is defined for the camera network: local layer and network layer. The local layer for each camera consists algorithms that run locally on each camera, its operation is invisible to the rest of the network. The network layer, which consists of predictive forwarding (Section 6.3.3) and probabilistic matching (Section 6.3.4) makes decisions using the output of many different local layers. Note that in the proposed distributed camera network, the network layer is a conceptual layer which is also executed on the smart cameras. With this concept in mind, the following section will describe the design and operation of the two layers.

6.4.1 Local Layer

The smart camera platform used in this work (Section 2.1) is capable of carrying out some simple computer vision tasks, including object tracking. The local tracking algorithm is briefly described here. To detect and track objects while ignoring non-relevant changes to the background, the approach proposed in [48, 107] is employed. The Mixture of Gaussian technique is used to segment moving objects from the background on a per-pixel basis. Then the foreground pixels are grouped into blobs representing moving objects. For people detection, head positions are located in the image using vertical projection histograms as described in [48]. This is easily accommodated for cameras that are mounted horizontally at human height. For ceiling mounted cameras, projective transformations can be used so that 3D vertical direction of persons standing on the ground plane will always map to 2D vertical lines in the transformed image [125]. An ellipse is fitted to each of the detected blobs in order to filter out those with an unrealistic width-to-height ratio [129]. Last, objects are continuously linked from frame to frame by a simple Kalman filter.

6.4.2 Interaction Between Local and Network Layer

The proposed distributed tracking network is comprised of two layers: the local layer (also referred to as intra-camera protocols) and the network layer (also referred to inter-camera protocols) as described in Section 6.3. The interaction between them can be described as follows: as the local detection algorithm successfully detects an object entering the FoV of a particular camera, the camera's inter-camera protocols (Section 6.3.4) are initiated to determine if the observation matches any of the received observations from its neighbouring cameras. If the matching procedure is unsuccessful, a new ID is assigned to the object. If

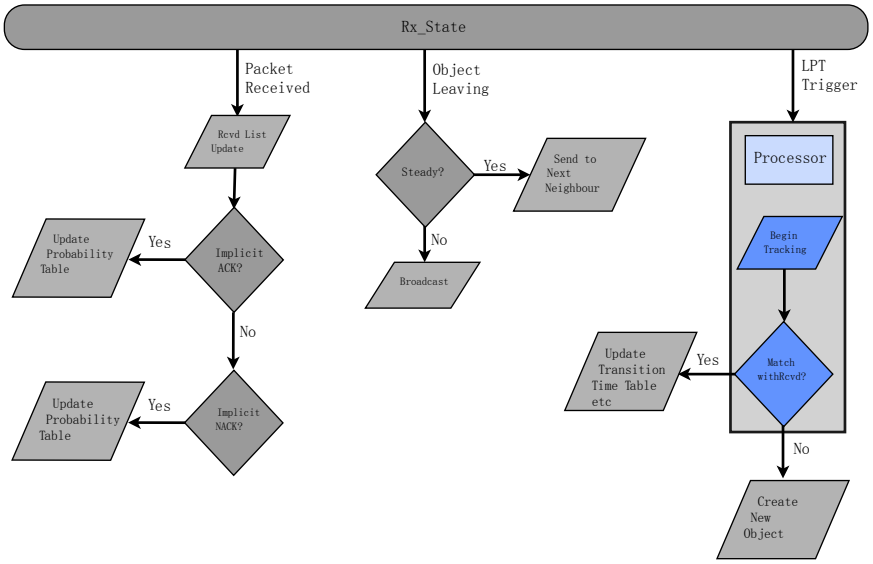


Figure 6.2: Camera state transition.

successful the matched ID is used to represent the object. In either case the camera's local tracking algorithm starts to track the object and obtains a reliable estimation of the key parameters such as colour distributions, time of departure, width-height ratio, speed of motion (requires pre-calibrated homographies, see Chapter 5) etc. As the object leaves the FoV, inter-camera protocols are initiated to forward the object description to the 'next' cameras to allow continuous tracking across cameras (see Section 6.3.3). This process repeats for each object that enters the FoV of a camera, enabling the objects to be tracked continuously in the network. This system would be infeasible without local processing algorithms or network level protocols.

Figure 6.2 is a flow-chart of the various states of a smart camera from a system

perspective. It is assumed that each camera has an in-built Low Power Triggering (LPT) mechanism (e.g. multiple Passive Infra-Red (PIR) sensors) which determines the occurrence of an event of interest before involving more higher-cost computation components (such as the DSP). The triggering mechanism allows the camera to remain in a very low-power state for the majority of the time and thus saving energy. Each camera is initialised in Rx state and the state changes according to one of the three possible events: object enters the FoV (LPT Trigger), object leaves the FoV (Object Leaving) and an object description is received from neighbouring camera (Packet Received).

The transition probabilities are initially estimated through an initialisation phase. During the initialisation phase, every detected object is matched to a list of received object descriptions using the appearance similarity with a threshold that is able to ensure a high true positive. The camera then broadcast the object to all of its one-hop radio neighbours when the object leaves its FoV. Once M objects have been successfully identified at a particular camera, the forwarding table is assumed to have reached a steady-state. From this point, only the cameras in the succeeding set are informed of the detection of an object. The transition probability is then updated using a sliding window over every M objects.

6.5 Evaluation

6.5.1 Simulation Framework

The proposed approach to the problem of object tracking in a distributed wireless camera network is validated by means of simulation, which is described in this section. Simulation of a camera network requires modelling objects moving in the sensor field. The simulation is therefore split into three major components:

the generation of a set of paths, the simulation of objects moving through the sensor field, and the operation of the camera network.

Path

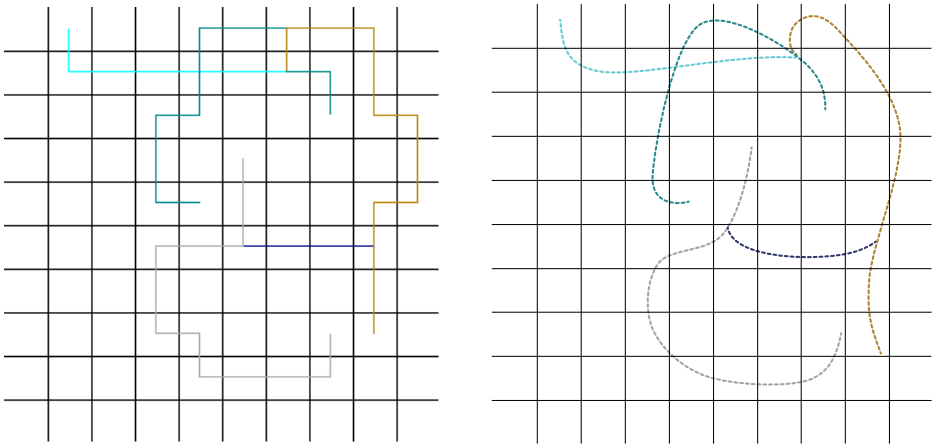
The developed simulation creates a deployment area which is divided into cells of a certain size, both of which are programmable at runtime. The simulation utilises the idea of a graph that contains vertices and edges. The vertices are located at the centre of each cell and the edges are links between each pair of vertices. The simulator randomly creates a number of *paths* consisting of a set of vertices connected with edges.

Figure 6.3(a) represents a typical deployment area with paths drawn in different colour. Although the paths have been restricted to be vertical and horizontal because of simplicity, the simulated camera network does not rely on the direction of path, instead, it is based on the connectivity of cameras. The fact that objects on the simulated paths move horizontally or vertically is *not* used. One equivalent map with realistic paths is shown in Figure 6.3(b).

Objects

The simulation also generates a programmable number of objects that move across the grid following randomly chosen paths. Each object appears in the network at a random time instance and at a random vertex. The object follows a path for a fixed duration. When an object reaches a vertex with multiple coinciding paths, its tendency is to remain on its current path with a small probability of changing paths.

It is difficult to model the appearance of an object in a simulated environment. As



(a) Generated paths (b) Equivalent paths

Figure 6.3: Simulated deployment area with paths drawn in different colour

noted earlier, the proposed approach uses colour histogram (H_c) to describe the appearance of the objects, and determine a similarity score using the similarity function $sim(H_{c1}, H_{c2})$. Unlike some of the other publications (e.g. [55]), this work does not assume a 100% matching between camera observations. It is the aim of the proposed approach to use the network level protocols to improve the system performance introduced by imperfect appearance based matching. To simulate the matching of appearance between objects, the simulator first assigns a random colour from 10 available colours to each object and then generates a colour matching score probabilistically based on distributions for cases when the two objects are of the same colour or not. For example, if two different objects have the same colour, their colour matching scores may follow a Gaussian distribution of mean of 0.8 and standard deviation of 0.1. A number of Gaussian distributions are evaluated which have varying parameters as shown in Table 6.1.

Table 6.1: Distributions of similarity scores. $N(\mu, \sigma)$ stands for a Gaussian distribution with mean μ and standard deviation σ. The decision threshold is the intersection of the two Gaussian distributions in each case.

Case no.	Distribution for identical colours	Distribution for different colours	Decision threshold
1	1 (constant)	0 (constant)	0.5
2	$N(0.8, 0.1)$	$N(0.2, 0.1)$	0.5
3	$N(0.8, 0.1)$	$N(0.4, 0.2)$	0.634
4	$N(0.8, 0.2)$	$N(0.2, 0.2)$	0.5
5	$N(0.7, 0.2)$	$N(0.3, 0.2)$	0.5

Camera Network

The simulator places a camera at each vertex on a path. The cameras track any objects that enter their cell and transmit the object's descriptive information to other cameras based on the approach described in Section 6.3.3. Upon first detecting an object, a match is sought from a list of potential candidates as described in Section 6.3.4 to allow objects to be tracked across the whole camera network. The duration of each object being tracked by a camera is equal to half of the time taken for the object to travel across the cell.

6.5.2 Performance Metric

Each new object is assigned a globally unique ID by the first camera that detects it, which in an ideal case, is carried in a sequential manner by each camera along the path as the object moves. However, for non-ideal cases, a number of scenarios can occur due to erroneous discrimination of objects. For example, a new object A, may not be assigned a unique ID as it may be recognised as an existing object B, who is in transition between two cameras. When B re-enters the network, it is likely to be recognised as another object C or detected as a new object that

the network has not seen before. There is no single metric that can be used to benchmark the system performance. Therefore three different metrics are chosen:

- *Object miss* occurs when a new object is not a assigned a new global ID;
- *ID loss* refers to the case when an object's ID is lost and
- *ID switch* refers to a globally unique ID being switched to a different object.

Although these metrics are different they are not completely independent and must be examined together. The three metrics are combined as a point in R^3 and the Euclidean distance is used as a comparative measure between different approaches.

6.5.3 Experiments

For the experiments, human beings have been chosen as the tracking targets across the simulated camera network. Assumptions made therefore correspond to this choice, although it should be noted that the simulator is capable of representing many other object types also. A deployment area of $300 \times 300\,\mathrm{m}^2$ has been chosen. This square was divided into grids of $30 \times 30\,\mathrm{m}^2$ cells. The number of paths was set to 5 and the number of vertices to about 12 per path. The distance between adjacent vertices for the experiments was 30 m and we assumed the transition time of objects between adjacent vertices to be a Gaussian distribution with mean $\mu = 25$ seconds with an standard deviation of $\sigma = 2$ seconds; this is based on the assumption that humans move with an average speed of 1.3 m/s.

Three approaches were compared in this experiment: The basic approach, the approach with only predictive forwarding, and the approach that combines pre-

Table 6.2: Different approaches compared. Colour histogram based method determines a match if similarity of two histograms are greater than a predefined threshold, approach 3 is proposed in Section 6.3.

Approach	Forwarding Tech.	Matching Tech.
1	Broadcast to all radio neighbours	Colour hist. based
2	Predictive forwarding	Colour hist. based
3	Predictive forwarding	Prob. matching

dictive sending and probabilistic matching. Cameras using the basic approach will always broadcast the object's information to their 1-hop radio neighbours and the matching of different observations is based on a detection threshold of similarity measure. The second approach is similar to the first, except that it only sends the object's information to the set of cameras that have a certain probability of being on the correct path (see Section 6.3.3). The decision of matching between observations was also based on a predefined threshold. The third approach was the proposed approach which differs from the second approach in that it makes a matching decision based on maximising a posterior probability as described in Section 6.3.4.

In all the experiments conducted, the network was simulated to run for one hour, in which there was a total of 300 objects moving through the camera network. Since each of them moved through the FoVs of about 10 cameras, they generated, in total, around $3,000$ observations in the camera network. The task of tracking is to correctly separate them into sets of observations that belongs to each individual.

There are two sets of experiments conducted:

Experiment 1

The aim of experiment 1 was to compare the proposed approach with the alternative approaches in various scenarios outlined in Table 6.1. Therefore three identical camera networks were simulated with each implementing a different object tracking strategy.

Experiment 2

To further analyse the behaviour of the proposed tracking strategy as time increases (which is identical to the increase in people count as the number of people that appears in the region during a fixed time interval was set to a constant), the simulation was repeated 10 times for each of the three approaches. The scenario presented in case 3 of Table 6.1 is assumed.

6.5.4 Simulation Results and Discussion

The performance of the approaches, outlined in Table 6.2, were compared in terms of the metrics of object miss, ID loss and ID switch. For each approach, the response of the network to different distributions of the similarity score (summarised in Table 6.1) of the colours of two objects were analysed and reported. The results are shown in Table 6.3. For case 1, even though objects of the same colour were always correctly matched whereas objects of different colours were always discriminated, due to the appearance of two objects of the same colour, none of the approaches can achieve a 100% correct tracking. This in fact represents a best case scenario and therefore an upper bound on the results, with all other parameters remaining the same.

Table 6.3: Experiment 1. Performance of the three approaches (Table 6.2) under different assumptions about the distribution of similarity score (Table 6.1).

Case no.	Approach 1			Approach 2			Approach 3		
	Obj miss	ID loss	ID switch	Obj miss	ID loss	ID switch	Obj miss	ID loss	ID switch
1	79	439	461	32	253	226	8	25	23
2	65	487	474	28	260	230	6	36	24
3	44	1124	999	20	970	722	6	127	69
4	55	776	698	22	652	464	5	211	42
5	58	1368	1259	19	1233	949	6	376	150

The remaining cases represent a number of more realistic possibilities. The difference in the results between approaches 1 and 2 in Table 6.3 shows that by employing the predictive forwarding strategy, the performance is improved by an average (over the three metrics) of between 32 - 51% for the 5 cases. Predictive forwarding limits the forwarding of object descriptions to the most likely nodes, which effectively reduces the set of candidates known by the subsequent cameras. Since an object is matched to one of the many candidates when it is re-detected by a subsequent camera, the likelihood of correct matches is increased when the number of candidates is reduced. Comparing approaches 2 and 3, it can be seen that the probabilistic matching procedure reduces the error by an average (again over all three metrics) of between 74 - 85% for the 5 cases. The improvement is a result of incorporating transition time, appearance and arrival probabilities in the matching stage.

To compare the results from another perspective, the three metrics were considered to be points in R^3 space and the Euclidean distances of each point from the origin, i.e. the magnitude of the representative vector were calculated. The results are shown in Figure 6.4 where it can be seen that approach three offers significant improvements over approaches 1 and 2 in all 5 cases.

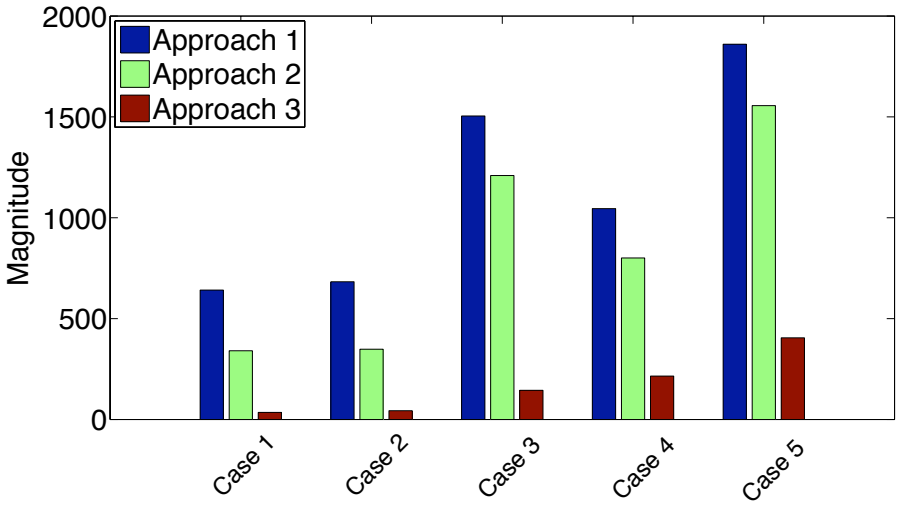


Figure 6.4: Experiment 1. Summary of results from Table 6.3 using the Euclidean distance as a measure.

Each time an object leaves the FoV of a camera, the observation is transmitted to another camera in order to continuously track the object. Figure 6.5.4 highlights the difference in the number of object descriptors transmitted in the whole network for the scenario presented in case 3 of Table 6.1). It can be seen that the number of transmissions is reduced significantly (approximately a 42% reduction) from approaches 1 and 2, which is due to the use of predictive forwarding strategy. When comparing approach 3 with the simple forwarding strategy in approach 1 this reduction is approximately 67%.

Figures 6.6(a), 6.6(b) and 6.6(c) show the increase in counts of object miss, ID loss and ID switch as the number of people entered the network increases for all three approaches in the scenario presented in case 3 of Table 6.1. Error bars in the figures were twice of the standard deviations for each metric which were computed over 10 identical repetitions of the simulation. For all three approaches, it is evident that there exists linear relationships between each metric and the number of people entering the region. This linear relationship is important in

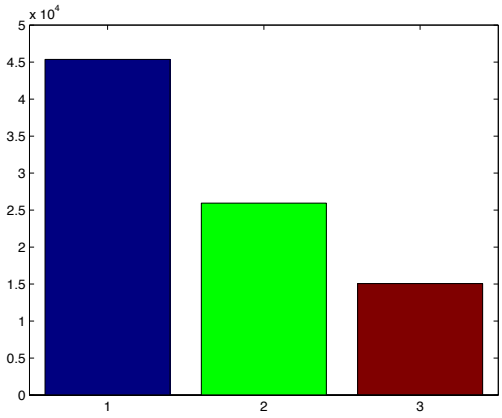


Figure 6.5: Experiment 1. Number of observations transmitted for case 3.

terms of the practicality of the proposed approach: Not only it is much more accurate than the alternatives, it is also able to maintain a constant error rate per time interval, indicating the system can be left on for much longer than the simulation duration.

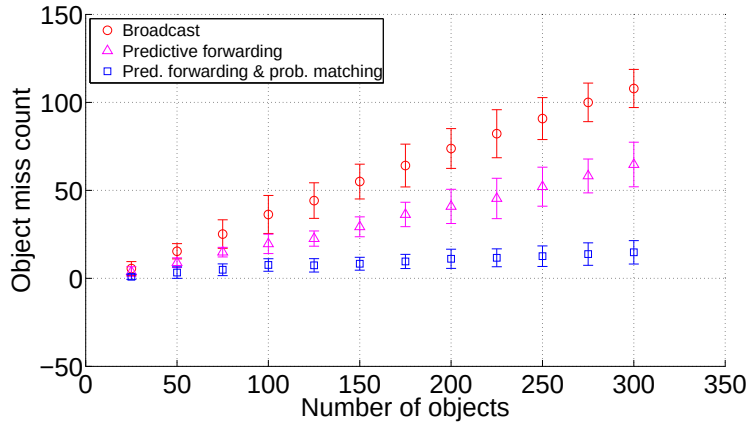
6.6 Chapter Summary and Future Work

In this chapter, a distributed object tracking design for wireless smart camera networks has been proposed.

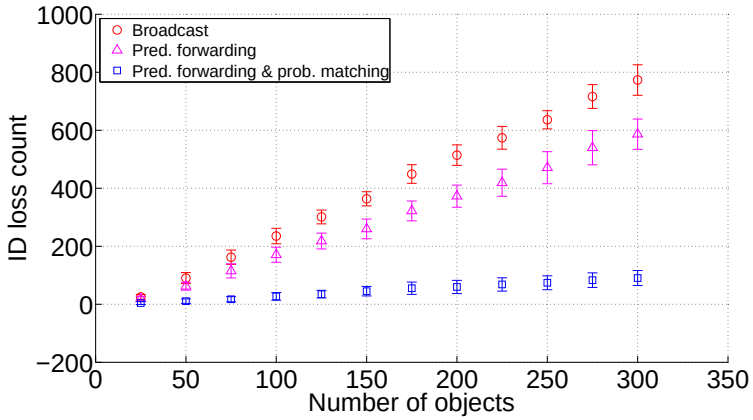
The network level algorithm proposed contains two stages. The first stage predicts the directions of motion of an observed object, allowing the object's information to be sent *only* to the set of cameras that have a high probability of subsequently detecting the object, thereby reducing the total number of observations transmitted. The second stage is termed probabilistic matching, which establishes the correspondence between a newly detected object to its location history from a list of possible candidates through a probability maximisation process. It has

been shown that with the integration of spatio-temporal cues, the proposed system performs significantly better (in terms of object-miss, ID-loss and ID-switch) than solely using the appearance to associate object observations. Due to the distributed nature of the system and the fact that cameras only require local knowledge for tracking, the proposed system is highly scalable. Thus, objective 3 in Section 1.2.2 has been fulfilled.

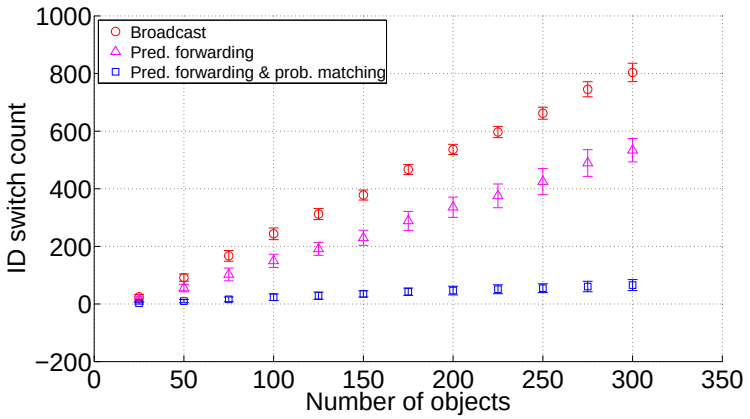
In addition to fully implementing the system in a real environment, as part of the future work, the assumed distributions will also be more closely examined. These include the Gaussian distributions used to model the arrival time and matching score. More spatio-temporal cues such as size, height-width ratio, object speed can also be integrated to improve the local and global tracking accuracy.



(a) Object miss count



(b) Id loss count



(c) Id swap count

Figure 6.6: Experiment 2. Performance of the three approaches as the number of people entered the region increases.

Chapter 7

Statistical Determination of Optimal Camera Configuration

7.1 Introduction

Networks of cameras have been widely used in the area of intelligent video surveillance (IVS). Based on user defined policies, IVS systems can automatically identify potential risks by detecting, localising, tracking and recognising targets and/or events of interest. For these camera networks, it is of vital importance that the optimal camera configuration (i.e. optimal location, orientation etc.) is determined before cameras are deployed, as the cost of modification can be expensive and the optimal configuration may provide saving on the total number of cameras used to achieve the same level of utility.

However, camera placement problem is not simple. Even in its simplest setting where an optimal configuration is sought to achieve a pre-defined coverage while minimising the number of cameras needed, the problem many be infeasible if

simple enumeration and search techniques are employed (e.g. [33, 56]) due to its NP-hard nature. Thus these methods can hardly be extended to large scale problems.

To address this issue, this chapter will first introduce a generalised statistical framework for the problem of selecting the optimal camera configuration, considering a number of user constraints and unknown number of cameras. Second, a Trans-Dimensional Simulated Annealing (TDSA) algorithm will be described which can effectively estimate the optimal number of cameras as well as the optimal parameters for these cameras. For small scale problems, the proposed approach offers similar very similar solutions to the optimal ones produced by Binary Integer Programming (BIP). For larger scale problems where BIP is clearly infeasible, it will be demonstrated that the proposed approach offers notable improvements over two alternative heuristics proposed in recent papers [56] and [128].

7.2 Related Work

Current research into automatic camera placement techniques generally tackles two types of problems, depending on the objects to be monitored. The first type ([38, 85, 87] etc.) focuses on computing the camera placement for a set of cameras that monitor the same object. The goal is usually to reconstruct the object using multi-view inputs. The second type ([9, 33, 124, 128] etc.) focuses on the best strategy to spread the views of cameras so that an area is monitored. The approach described in this chapter belongs to the latter category.

The automatic camera placement problem has been studied by various authors in a number of contexts with different constraints and requirements. The earliest

related work was published by O'Rourke [88] who provided the formulation of the Art Gallery Problem and its solutions. The Art Gallery Problem is the assignment of guards to different positions in an art gallery in order to achieve the maximum visual coverage of the paintings. In essence, this is equivalent to finding the best locations for a set of infinite-depth omni-directional cameras to achieve the optimal coverage of the observation area.

Recently, Erdem and Sclaroff [33] formulated the general camera placement problem in an optimisation framework. Given the set of all constraints τ required to achieve a specific task γ, the problem is to find the optimum placement for a set of cameras $\mathbf{\Pi}$ in the area V satisfying τ and minimising a given cost function $G(\bullet)$. The authors proposed four instances of this general formulation and solved them using Binary Integer Programming (BIP) over a discrete problem space.

Erdem and Sclaroff's formulation has been adopted by Horster and Lienhart [56]. The difference between the two is that instead of restricting the areas to be polygon shaped, Horster and Lienhart used points to represent space. These points may have different importance distribution. Furthermore, to deal with the problem of scaling the BIP algorithm to handle large regions, heuristics are used to solve the problem within a reasonable time and moderate memory consumption. The compromise however is that global optima cannot be guaranteed.

Yao et al. [124] argued that only maximising visibility is insufficient for persistent and automated tracking. Therefore the authors proposed to incorporate a hand-off success rate (the percentage of successful hand-offs) analysis in determining camera placement, preserving necessary uniform overlapped field of views (FoVs) between adjacent camera for an optimal balance between coverage and hand-off success rate.

Various other requirements have been considered. Bodor et al. [9] proposed an

approach to the camera placement problem that tries to optimise the camera network's ability to observe a set of predefined tasks, such as human motion. The authors developed an analytical formulation of the observation problem, in terms of the statistics of the motion in the scene and the total resolution of the observed actions. An optimisation routine is used to find the location and orientation that optimise the observation criteria.

The observability of frontal faces has been incorporated in the process of finding the optimal camera placement by Ram et al. [97]. The authors showed the derivation of a performance metric to compute the probability of observing an object of random orientation from one sensor and used it to estimate the performance of multiple sensors. Motion sensors which provide location information can be included in the performance metric. Similar to all the other approaches, the performance metric is used as the objective function whose maximum is sought by employing an optimisation routine.

Zhao and Cheung [128] described a visibility model that takes in a number of realistic inputs: arbitrary-shape 3D environments, 3D camera models, occupant traffic models, self-occlusion and mutual occlusion. The visibility model is used in evaluating the objective - maximisation of the probability of tracking visual tags. The maximisation is done with two proposed Binary Integer Programming (BIP) algorithms as well as a greedy implementation designed to cope with the complexity of BIP. Similarly, Mittal and Divas [82] considered occlusion in a probabilistic manner in the visibility calculation and used an existing simulated annealing method to compute the optimum parameters of the cameras. However, it is not clear how the optimal number of cameras is determined, which will be addressed in this chapter.

Many of the existing methods formulated the problem as combinatorial optimisations [33, 56, 128] that belong to a class of NP-hard problems. Heuristics

methods have been developed with computational requirements proportional to small powers of N, i.e. in a relatively simple environment. However heuristics can be problem-specific as there is no guarantee that a heuristic will work for all setups. Alternatively the 'divide and conquer' approach can be taken to break-down the problem into a number of simpler ones. The problem with this approach is that the result is only optimal when the sub problems are disjoint, which is often not the case. To counter these problems, the camera placement problem will be formulated as a maximum a-posteriori model selection and optimisation in Section 7.4 and a viable solution using the trans-dimensional simulated annealing will be described in Section 7.5.

7.3 Problem Definition

The camera model used in this work is not overly restrictive. In fact each camera $c_i = \left[p_1, p_2, ..., p_{n_p}\right]^\top$ consists of n_p number of parameters which may include the camera location in x, y, z directions of the area to be monitored, orientation, tilting angle, lens type, price etc. The camera parameter space is denoted as $\mathcal{C} = \mathcal{R}^{n_p}$. Note that although c_i is a vector, it is not boldfaced in order to avoid confusion in subsequent sections of this chapter.

Although there is no particular restrictions on what parameter to include in the camera model, it is however essential that given a particular network of cameras $\boldsymbol{\theta} = [c_1, c_2, ..., c_{n_c}]^\top$, a set of constraints $\mathbf{r} = \{r_j \mid r_j \in \{r_1, r_2, ..., r_{n_r}\}\}$ and a description of the area to be monitored ξ, there must exist some function $L(\mathbf{r} \mid \boldsymbol{\theta})$ that determines how well the constraints $\mathbf{r}$ are jointly satisfied by $\boldsymbol{\theta}$. For example, if it is required to achieve a frontal face capturing rate of 80%, but a particular set of cameras can only achieve 60%, then it may be said that the cameras have achieved $60/80 = 75\%$ of the requirement.

Given the camera parameter space $\mathcal{C}$, an input environment ξ, a set of user requirements $\mathbf{r}$, the camera placement problem can be defined as the selection of the camera configuration that meets $\mathbf{r}$ while minimising the number of cameras used. It will be shown in Section 7.4 that this formulation derived can be easily applied to the problem where some utility is to be maximised while keeping a constant number of cameras.

7.4 Generalised Framework

To formulate the problem in a Bayesian modelling context, suppose that there are a countable collections of candidate models $\mathcal{M}_k$ indexed by a model indicator $k \in \mathcal{K}$. Each model $\mathcal{M}_k$ has an $\|\mathcal{M}_k\| = n_k$ dimensional vector of parameters $\boldsymbol{\theta}_k$. Note that for the sake of simplicity the model indicator is used to represent the model. For example, the dimension of the model is $\|k\| = n_k$. In the context of camera placement, the model of a camera network configuration is reflected by the number of cameras in the configuration and *each camera in the configuration is considered as a random variable* over the space of $\mathcal{C}$.

The camera placement problem is defined by the joint posterior,

$$\phi_{opt} = \arg\max_{\phi_k \in \mathcal{X}} \left\{ p\left(k, \boldsymbol{\theta}_k \mid \mathbf{r}\right) \right\}, \tag{7.1}$$

where $\phi = (k, \boldsymbol{\theta}_k)$ denotes a camera configuration which contains a model indicator k as well as camera parameters of the model $\boldsymbol{\theta}_k$. The joint state space is thus $\mathcal{X} = \bigcup_{k \in \mathcal{K}} \left(\{k\} \times \mathcal{C}^{n_k} \right)$. This formulation can be considered as: Given there is an observation that the list of constraints and requirements have been satisfied, the problem is to find the optimal model k and the optimal parameters $\boldsymbol{\theta}_k$ that are most likely to have led to this observation. For example, if the requirement is covering the maximum amount of the floor with a given number of

cameras. Equation (7.1) can then be interpreted as finding the most likely camera configuration that has caused maximum coverage to be observed (satisfied).

Expanding Equation (7.1) using Bayes theorem,

$$\phi_{opt} = \arg\max_{\phi_k \in \mathcal{X}} \left\{ L\left(\mathbf{r} \mid k, \boldsymbol{\theta}_k\right) p\left(\boldsymbol{\theta}_k \mid k\right) p\left(k\right) \right\}. \tag{7.2}$$

The first term $L\left(\mathbf{r} \mid k, \boldsymbol{\theta}_k\right)$ is the probability of satisfying the requirements and constraints $\mathbf{r}$ by the given set of camera parameters $\boldsymbol{\theta}_k$ and is therefore called the likelihood. The second term $p\left(\boldsymbol{\theta}_k \mid k\right)$ is termed the parameter prior since it defines the prior probability of the set of camera parameters. The prior term allows the user to set preferences on the parameters of the cameras. For example, wall locations can be preferred through assigning high prior to cameras that are located on walls and omni-directional cameras may be unwanted by associating them with low prior. The last term $p\left(k\right)$ is the model prior which captures user preference on the models. In the most settings where all cameras are treated equally, this term can be dropped. This is the case for all the experiments presented in Section 4.5.

The formulation (Equation (7.2)) can be converted to a penalised model selection problem for further investigation. An equivalent form of Equation (7.2) can be obtained as,

$$\phi_{opt} = \arg\min_{k, \boldsymbol{\theta}_k} \left\{ -\log\{ L\left(\mathbf{r} \mid k, \boldsymbol{\theta}_k\right) p\left(\boldsymbol{\theta}_k \mid k\right) \} + \log\{1/p\left(k\right)\} \right\}. \tag{7.3}$$

If letting $p(k) = \exp\{-n_k\}$, this becomes the well known AIC [3] that is often used in model selection problems,

$$\phi_{opt} = \arg\min_{k, \boldsymbol{\theta}_k} \left\{ -\log\{ L\left(\mathbf{r} \mid k, \boldsymbol{\theta}_k\right) p\left(\boldsymbol{\theta}_k \mid k\right) \} + n_k \right\}. \tag{7.4}$$

The above formulation is designed for the problem when the constraints are to be met to a particular level while minimising the number of camera required. If however the problem is to maximise an utility with a fixed number of cameras, the model indicator can simply be dropped out of the formulation to keep a constant model dimension.

In some cases one may be interested to minimise the total cost of a camera network when there are more than one types of cameras with different prices available. In these cases, the model prior can be assumed uniform and then the penalty term (n_k in Equation (7.4)) can be replaced with a function on the total price of the particular configuration.

7.5 Trans-dimensional Simulated Annealing

The trans-dimensional simulated annealing [4, 12] is used to solve the stochastic problem posed in Section 7.4. Simulated Annealing (SA) is a class of algorithms capable of locating good near-optima of objective functions in large search spaces. The term simulated annealing derives from the interesting observation that as a heated material slowly cools down, its molecules will line up in a rigid pattern corresponding to a state of minimum energy provided that the cooling process is sufficient slow. SA algorithms mimic this process and have been proven to converge [45]. Trans-dimensional simulated annealing (TDSA) is a class of algorithms that extend the traditional simulated annealing by allowing moves that not only change the parameters of the model but as well move between plausible models. Therefore, TDSA algorithms are able to locate the models and parameters that minimise objectives such as AIC [3], BIC [104] and MDL [100]. This section details the design of a such algorithm which can be used effectively for large scale camera placement problems.

To start the derivation process, a slight modification to AIC in Equation (7.4) is introduced to allow the control of severity of the penalty placed on model order,

$$J(\phi) = -\log\left\{L(\mathbf{r} \mid k, \boldsymbol{\theta}_k)\, p(\boldsymbol{\theta}_k \mid k)\right\} + \gamma n_k. \tag{7.5}$$

Given the objective function $J(\phi)$ to be minimised over the joint space $\bigcup_{k\in\mathcal{K}}(\{k\} \times \mathcal{C}^{n_k})$, the corresponding Boltzmann distribution [67] can be defined as,

$$\begin{aligned} b_T(\phi) &\propto \exp\left\{-J(\phi)/T_i\right\} \\ &= \left(L(\mathbf{r} \mid k, \boldsymbol{\theta}_k)\, p(\boldsymbol{\theta}_k \mid k)\right)^{1/T_i} \exp\{\gamma n_k/T_i\}, \end{aligned} \tag{7.6}$$

where T_i is a decreasing cooling schedule with $\lim_{i\to\infty} Ti = 0$. There exists many valid types of cooling schedules, such as linear, logarithmic and geometric schedules. In particular, the logarithmic schedule has been proven to always lead to convergence but at very slow rate. For real camera placement problems, near-optimal solutions are often acceptable. Therefore, the geometric schedule with initial value of T_0 has been chosen, as based on a few pilot runs of the experiment, this schedule leads to a balanced speed and accuracy of convergence.

$$T_i = \rho T_{i-1}, \tag{7.7}$$

where $0 < \rho < 1$ is the cooling coefficient.

The TDSA algorithm proposed involves simulating a non-homogeneous Markov chain whose stationary distribution at temperature T_i is proportional to $b_T(\phi)$. For each temperature, given that the Markov chain is at some current state ϕ, the algorithm first selects a move type m from a set of predefined moves, which are birth m_b, death m_d and update m_u with prior probability $p_m \in \{p_b, p_d, p_u\}$. It then generates a new candidate camera configuration ϕ' by sampling an auxiliary variable $\mathbf{u}$ from a known density $g_m(\mathbf{u})$. $\mathbf{u}$ is subsequently combined with the

current state ϕ through some deterministic function h: $\phi' = h(\phi, \mathbf{u})$ to produce the proposed candidate. The move can either change the dimensionality of the state (i.e. jump to a different model) or the chain can remain at the current model (i.e. dimension remains unchanged). Last, ϕ' is accepted with probability $\alpha(\phi, \phi')$ [46], where

$$\alpha(\phi, \phi') = \min\left\{1, \frac{b_T(\phi')p'_m g'_m(u')}{b_T(\phi)p_m g_m(u)} \left|\frac{\partial(\phi', u')}{\partial(\phi, u)}\right|\right\}, \tag{7.8}$$

Here p'_m is the probability of choosing the reverse move of m and g'_m is the associated known proposal density of the reverse move. For the proposed algorithm, birth, death and update moves have been selected. The birth and death constitute a pair of reversible moves that allow the state of the chain to grow from k to $k+1$ and decrease from k to $k-1$. The reverse move of the update move is itself. Although other moves, typically merge and split, can be defined, the ones selected have been tested and found to produce satisfactory results. Each move will be described in more details in the following sections.

7.5.1 Birth and Death Moves

Birth and death moves are a pair of reversible moves that facilitate model dimension changes. The birth move proposed is rather simple: for a given state ϕ, a new camera is created randomly and added to ϕ to form ϕ'. Similarly the death move is achieved by randomly removing a camera from the existing camera configuration. The acceptance ratio of the moves are,

$$\alpha_{birth} = \min\left\{1, \frac{b_T(\phi')p_{m_d} n_{max}}{b_T(\phi)p_{m_b}(n_k+1)}\right\}, \tag{7.9}$$

$$\alpha_{death} = \min\left\{1, \frac{b_T(\phi')p_{m_b}n_k}{b_T(\phi)p_{m_d}n_{max}}\right\}, \tag{7.10}$$

where n_{max} is a user defined maximum allowable dimension of the models. The Jacobian term of Equation (7.8) for both birth and death moves can be shown to be 1. The term n_{max} was originally derived to be $(n_{total} - n_k)$, where n_{total} is the total number of candidate cameras from which an optimal subset is to be selected (more details in Section 7.6.1). The inverse of this term is the probability of proposing a new camera which is then added to the existing cameras to facilitate the birth move. For large scale problems, n_{total} is a very large number compared to n_{max}. The problem with this original form is that all model spaces may be explored. In camera placement problems, this is rarely necessary as, for example, if there are 5000 candidate cameras, the model dimension of the optimal solution will always be much less than 5000 (but the theoretical maximum dimension size is 5000). Therefore to speed up the optimisation process, n_{max} is used to replace $(n_{total} - n_k)$ so that there is a lower chance of exploring models of high dimensions. When n_{max} is passed (i.e. $n_k > n_{max}$), n_{max}/n_k in Equation (7.9) starts to decline (note that if using the originally derived term this ratio will not decline), thus reducing the acceptance ratio of moves that further increase model dimensionality. Therefore, n_{max} can significantly reduce the optimisation time as it eliminates the unnecessary computation of the $J(\phi)$ when the dimension of ϕ is larger than necessary. The optimisation result is not quite sensitive to the value of n_{max} as long as it is slightly smaller, equal or larger than the optimal model dimension. In case when this prior knowledge about the optimal model dimension is not available, a relatively large value can be set. This is the case for all experiments conducted in Section 7.6. This benefit is only evident when the total number of candidate cameras is large (e.g. hundreds, thousands). When it is small, n_{max} does not have any effect as the computation of $J(\phi)$ is quick, and the optimisation can explore the all the model spaces even though some of these are unnecessary.

7.5.2 Update Moves

The update move is an important move that allows the estimation of better camera configuration while preserving the dimension of the state. It starts by first randomly selecting an existing camera from the current state of the configuration, i.e. select c_i from $\boldsymbol{\theta}_k$ and then update this camera with random walk. For example, if a camera is described by its location on a planar map and its orientation, the random walk may be constructed as a consecutive Gaussian perturbations of the x location, the y location and the orientation, with means being their current values and some pre-defined standard deviations. Since the random walk is symmetrical and the move does not involve dimension changes, the acceptance probability is reduced to,

$$\alpha_{update} = \min\left\{1, \frac{b_T(\phi')}{b_T(\phi)}\right\}. \tag{7.11}$$

7.5.3 Summary

The steps of the proposed trans-dimensional simulated annealing algorithm for determining the optimal number of cameras as well as the parameters of each camera are summarised in Algorithm 7.1. It can be seen from Algorithm 7.1 there is exactly one evaluation of the objective function $J(\phi)$ in each iteration of the Markov chain and all the rest of the computation takes constant time. Therefore the speed of the proposed algorithm is almost proportional to the speed of evaluation of J, which is case dependent.

Algorithm 7.1: Trans-dimensional simulated annealing for determining the optimal camera configuration.

7.1.1 Function $\{\phi_{opt}, s_{max}\} \leftarrow \text{TDSA}\,(T_0, T_e, p_{m_b}, p_{m_d}, \phi_0, l, \gamma, \rho)$
Input:
T_0, T_e – The initial temperature and the end temperature.
p_b, p_d, p_u – The probability of choosing the birth, death and update move respectively.
$\phi^0 = (k^0, \boldsymbol{\theta}_k^0)$ – The initial state of the Markov chain, where
k^0 – The initial model order.
$\boldsymbol{\theta}_k^0$ – The initial camera configurations.
l, γ, ρ – The chain length, model penalty parameter and cooling coefficient.
Output:
ϕ_{opt} – The optimal model and the optimal camera parameters.
s_{max} – The optimal value of the objective function Equation 7.5.
7.1.2 **begin**
7.1.3 $T \leftarrow T_0$,$\phi \leftarrow \phi_0$,$s_{max} \leftarrow -\infty$.
7.1.4 $s_c \leftarrow \text{J}\,(\phi, \gamma)$ /*Equation 7.5*/.
7.1.5 **while** $T \geq T_e$ **do**
7.1.6 **for** $j \in \{1, 2, 3, ..., l\}$ **do**
7.1.7 $\beta \leftarrow \text{RAND}\,(0, 1)$.
7.1.8 **if** $\beta \leq p_b$ **then**
7.1.9 $\phi' \leftarrow \text{BIRTH}\,(\phi)$ /*Birth Move*/.
7.1.10 $s_p \leftarrow \text{J}(\phi', \gamma)$ /*Equation 7.5*/.
7.1.11 $\alpha \leftarrow \text{ALPHA}_B(\phi', s_p)$ /*Equation 7.9*/.
7.1.12 **else if** $\beta \leq p_b + p_d$ **then**
7.1.13 $\phi' \leftarrow \text{DEATH}\,(\phi)$ /*Death Move*/.
7.1.14 $s_p \leftarrow \text{J}(\phi', \gamma)$ /*Equation 7.5*/.
7.1.15 $\alpha \leftarrow \text{ALPHA}_D(\phi', s_p)$ /*Equation 7.10*/.
7.1.16 **else**
7.1.17 $\phi' \leftarrow \text{UPDATE}\,(\phi)$ /*Update Move*/.
7.1.18 $s_p \leftarrow \text{J}(\phi', \gamma)$ /*Equation 7.5*/.
7.1.19 $\alpha \leftarrow \text{ALPHA}_U(\phi', s_p)$ /*Equation 7.11*/.
7.1.20 **end**
7.1.21 $\mu \leftarrow \text{RAND}\,(0, 1)$.
7.1.22 **if** $\mu < \alpha$ **then**
7.1.23 $\phi \leftarrow \phi'$, $s_c \leftarrow s_p$.
7.1.24 **if** $s_c > s_{max}$ **then**
7.1.25 $s_{max} \leftarrow s_c$, $\phi_{opt} \leftarrow \phi$.
7.1.26 **end**
7.1.27 **end**
7.1.28 **end**
7.1.29 $T \leftarrow \rho T$.
7.1.30 **end**
7.1.31 **end**

7.6 Evaluation

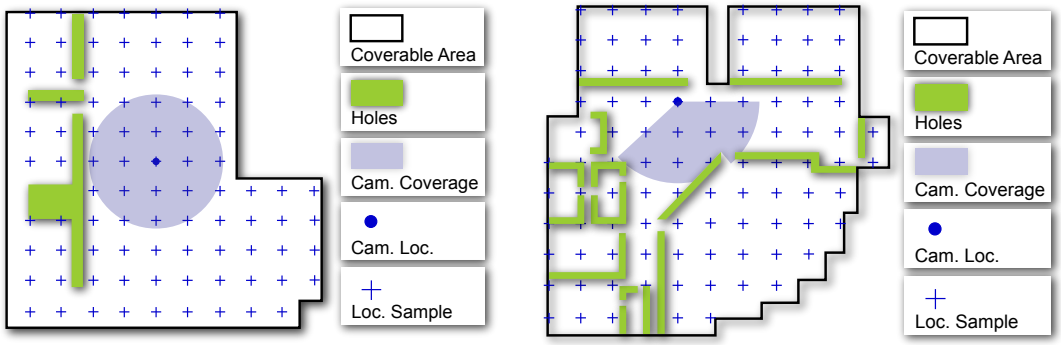
The generalised framework and the proposed TDSA algorithm are evaluated in a number of ways. The evaluation section starts with a description of the experiment methodology and then compares the proposed approach with Erdem and Sclaroff's method [33], the GREEDY algorithm by Zhao et al. [128] and the dual sampling proposed by Horster and Lienhart [56] in a number of setups. Note that Erdem and Sclaroff's methods is based on BIP which produces optimal results but only for small scale problems. The later two methods are customised heuristics designed to deal with this shortcoming of BIP but at the expenses of optimality. Last, the flexibility of the proposed approach is demonstrated through dealing with a more complex scenario where, apart from achieving 100% coverage, a number of pre-labelled critical areas are also required to be covered by at least two cameras.

7.6.1 Experimental Methodology

Since the focus is on establishing a generalised statistic framework for the selecting the optimal camera configuration and proposing an effective algorithm to deal with the problem of scalability of BIP, therefore a simple setup has been chosen to evaluated the proposed algorithm: Given a 2D map of an area and the camera specifications, the goal is to compute the optimal configuration that uses the least number of cameras and achieve 1). a desired total coverage of 100% (section 7.6.2) or 2). a desired total coverage of 100% and 100% coverage of the pre-labelled critical regions by 2 or more cameras (section 7.6.3).

Floor Plan

Two floor plans were used in the experiments which are shown in Figure 7.1. The first floor plan ($638 \times 616\text{pixel}^2$) is adopted from [33] and the second floor plan($804 \times 733\text{pixel}^2$) is a modification from the real floor plan of a typical university building. Both of the floor plans consist of only polygonal areas where camera coverage is possible and holes (cavities) where camera viewing frustums are blocked completely or partially.



(a) Floor plan A (b) Floor plan B

Figure 7.1: Floor plans used in the experiments.

Camera Model

Each camera is described by 4 parameters, $c = [x, y, o, t]$, which are x and y locations, orientation o and type t. The camera type parameter specifies the field of view and the depth (maximum distance visible in pixels) of the camera. For omni-directional cameras, orientation is irrelevant and the field of view is always 360°. The coverage area of a omni-directional camera and that of a PTZ or perspective camera is depicted in Figure 7.1(a) and Figure 7.1(b) respectively.

Since the input environment map is naturally available in the unit of pixel, there-

fore most units in the experiments were converted to pixels. In particular, these include camera locations and coverage area size.If given the real parameters of a camera, it not difficult to do so. For example, the OmniVision OV9655 has a pixel size of 3.18×10^{-6}m, focal length of $4.85 \times 10^{-}3$m. Thus the focal length in pixels is $4.85 \times 10^{-3}/(3.18 \times 10^{-6} = 1525$pixels. Assuming a perspective camera is mounted horizontally on a wall at human height, the distance at which the width of the image of a human head of 20×10^{-2}m will be exactly 50pixels is $= 20 \times 10^{-2}/(50 \times 1525) = 6.1$m. If the floor is 26×25m^2 and the size of the image of the floor plan is 638×616pixel^2, then each meter in real world corresponds to $638/26 = 616/25 = 24.5$pixels on the image of the floor plan. $6.1\text{m} \times 24.5\text{pixel/m} = 149.45$pixels. Therefore the maximum depth of the camera has to be at most 150pixels to ensure human heads are at least 50pixels wide on the images captured by the camera.

Sampling

Ideally, the camera parameters are continuous (except the camera type). A camera can be positioned anywhere in an environment and posed at any angle. However, due to the use of optimisation methods as opposed to closed form solutions, the parameters were *sampled* to reduce computation. The 2D environment maps were divided into grids to allow easier computation of cameras' coverage regions. The locations where cameras can exist were restricted to a number of location samples, which are represented as crosses in Figure 7.1(a) and Figure 7.1(b). Similarly, the orientations of the cameras (except omni-directional cameras) were also sampled. The sampling of parameters allows the construction of the set of *candidate cameras* (also sometimes referred to as the sampled set), from which a optimal subset is to be selected to satisfy the user constraint. All approaches were implemented on a Intel Core 2 Quad 2.5GHz PC with 4GB memory running

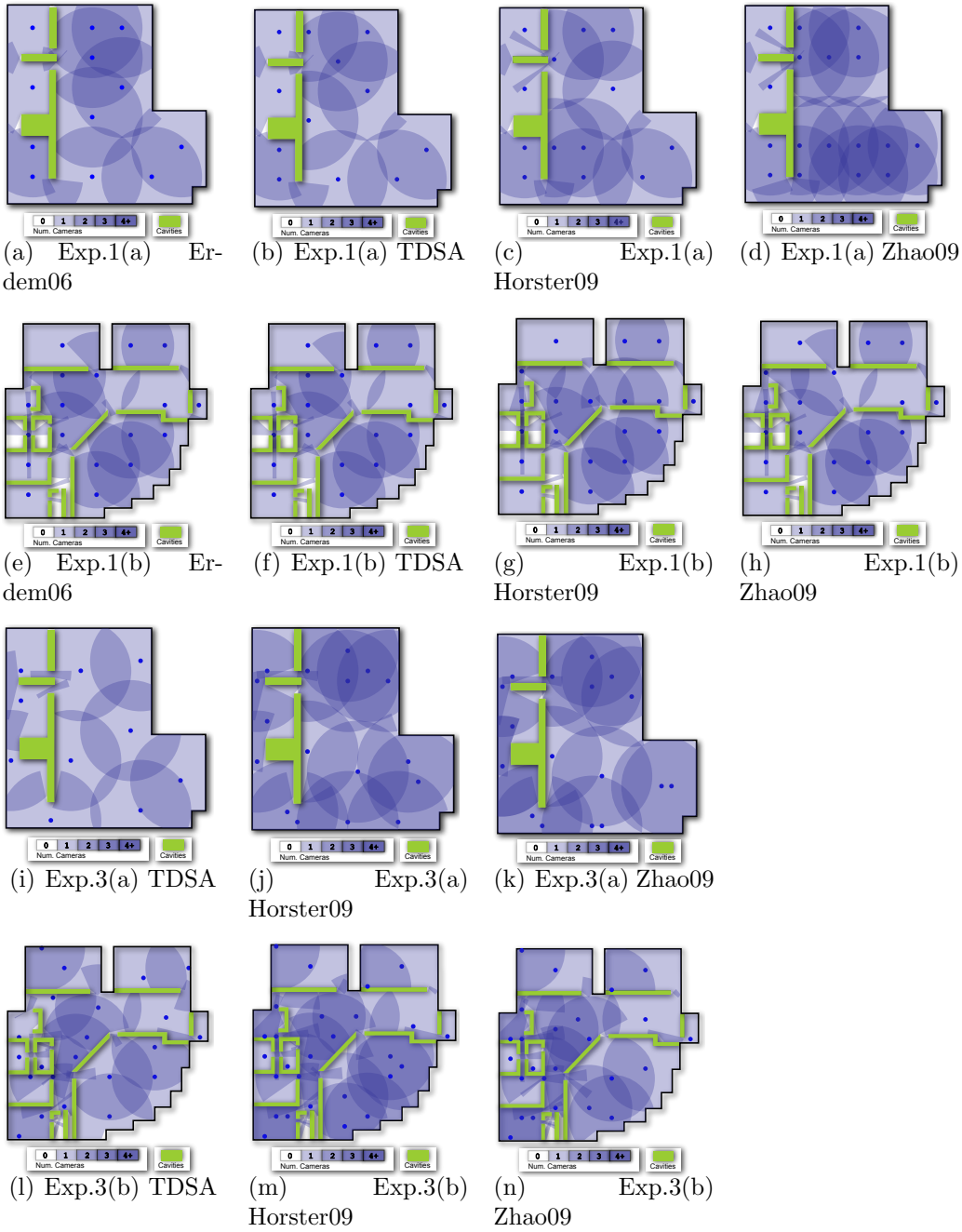
Matlab 2010b (64bit). For the BIP optimisation routine required in Erdem and Sclaroff's approach, the *binprog* function from Matlab's optimisation toolbox was used.

7.6.2 Performance Comparison

First, 4 experiments were conducted using floor plan A and B to evaluate the performance of the proposed approach against the alternative approaches: Erdem06 [33], Zhao09 [128] and Horster09 [56] under different setups. All the experiments conducted have the same objective: minimising the number of cameras used to achieve a 100% coverage of the two floor plans. To compute the likelihood (as required by Equation 7.5) of a particular camera configuration in satisfying the required constraints, the visibility computation routine described in [33] is used first to compute the coverage areas of all the cameras in the configuration, which is denoted as $\mathrm{COV}(\phi)$. Then the likelihood can be computed by $L(\mathbf{r} \mid \phi) \propto \exp\left\{-\left(\mathrm{COV}(\phi) - r_{cov}\right)^2 / 2\sigma^2\right\}$, where r_{cov} is the desired coverage percentage, which in this case is 100% and σ is set to 0.1.

The separation between each camera location samples were 90, 60 and 30 pixels in both x and y directions for the 4 experiments respectively. Two types of cameras were employed: omni-direction cameras with 360° field of view (FoV) and cameras with 120° FoV. Both types of cameras have depth of 150pixels and the orientation of the 120° FoV camera is sampled every 20°. The 4 experimental setups are summarised in Table. 7.1. For TDSA, the following parameters were used: $T_0 = 10$, $T_e = 10^{-4}$, $\rho = 0.99$, $\gamma = 10^{-5}$, $n_{max} = 200$, $p_b = p_d = 0.2$. The results are plotted in Figure 7.3(a) and Figure 7.3(b) and the computed placements strategies of Exp. 1 and Exp. 3 are shown in Figure 7.3.

The Matlab implementation of BIP (used in Erdem06) uses a branch and bound



(a) Exp.1(a) Erdem06 (b) Exp.1(a) TDSA (c) Exp.1(a) Horster09 (d) Exp.1(a) Zhao09

(e) Exp.1(b) Erdem06 (f) Exp.1(b) TDSA (g) Exp.1(b) Horster09 (h) Exp.1(b) Zhao09

(i) Exp.3(a) TDSA (j) Exp.3(a) Horster09 (k) Exp.3(a) Zhao09

(l) Exp.3(b) TDSA (m) Exp.3(b) Horster09 (n) Exp.3(b) Zhao09

Figure 7.2: Plot of the camera configurations computed by all 4 methods for Exp. 1 and 3.

Table 7.1: Experiments conducted for comparison of different approaches.

	Floor plan	Cam. loc sep.	Cam. FoV FoV	Orientation sep.	Depth	Num. candidate
Exp. 1(a)	A	90	360	NA	150	28
Exp. 1(b)	B	90	360	NA	150	28
Exp. 2(a)	A	60	360	NA	150	79
Exp. 2(b)	B	60	360	NA	150	64
Exp. 3(a)	A	30	360	NA	150	304
Exp. 3(b)	B	30	360	NA	150	251
Exp. 4(a)	A	30	120	20	150	5472
Exp. 4(b)	B	30	120	20	150	4518

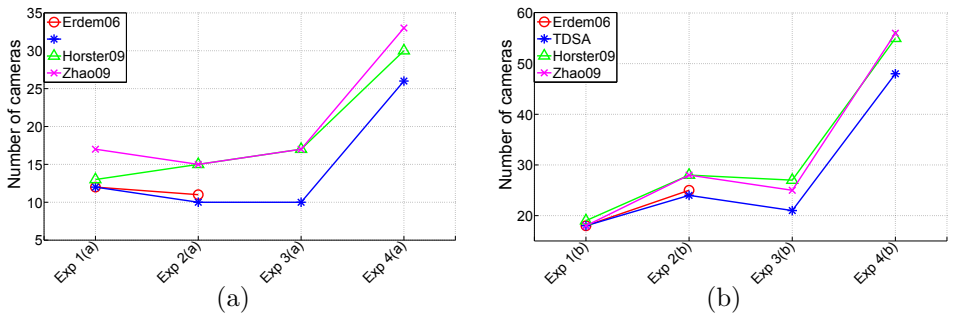


Figure 7.3: Number of cameras computed to cover floor plan A (a) and floor plan B (b). Note that Erdem06 was not able to produce results for Exp.3 and Exp.4.

algorithm which in the worst case scenario will visit all the possible combinations. In Exp.1 and Exp.2, the BIP algorithm took less than 10 seconds to complete but for Exp.3 and Exp.4, where the search spaces were approx. 2^{300} and 2^{5000}, it was unable to find a solution in feasible time (48 hours in this case). This severely limits the applicability of Erdem and Sclaroff's approach to larger scale problems. The proposed approach, on the other hand, takes a few hours for each experiment despite the rapid growth of search space size from Exp.1 to Exp.4 (Exp.4(b) took 5.5 hours on a 2.5GHz PC running matlab). It can be seen by inspecting Figure 7.2(a) and 7.2(b), Figure 7.2(e) and 7.2(f) that both Erdem06 and TDSA produced results that are almost the same, meaning optimal solutions have been found by TDSA. Comparing the proposed approach with the other 2 heuristics in Figure 7.2, it is evident that the results of Horster09 and Zhao09 are much more cluttered, especially in cases when the search spaces are relatively large (comparing Figure 7.2(i) with 7.2(j), Figure 7.2(k) with 7.2(l), 7.2(m), 7.2(n)). The same conclusion can be drawn by inspecting the two summary plots in Figure 7.4(a) and Figure 7.3(b), where the number of cameras used to cover each floor plan computed by all 4 methods are plotted. Overall, the strategy computed using the proposed TDSA methods requires 18.6% and 21.2% less cameras than Horster09 and Zhao09 respectively.

In addition, the proposed approach is more flexible as the formulation (Equation 7.4) penalises the increases in the number of cameras that do not bring much gain to the coverage. By comparing the results of Erdem06 and TDSA in Figure 7.3(a) and Figure 7.3(a), it appears that the TDSA out-performs the optimal BIP method(Erdem06) in Exp.2. This is caused by the fact that the proposed approach omitted a few pixels in the corners, achieving coverage of 99.99% as opposed to 100% achieved by the alternative approaches. This behaviour is primarily due to the penalty of the extra camera outweighed the 0.01% coverage increase in the objective function (Equation 7.5) and thus was not included in

the optimal set.

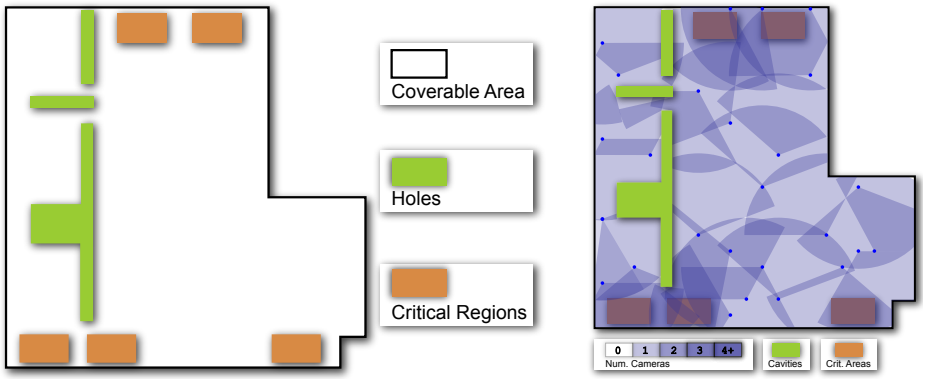
7.6.3 Example of A Different User Objective

The objective used in the previous experiments (Exp.1-4) is only a particular case of the various objectives that can be dealt with by the proposed formulation of the problem. This section shows the result of computing optimal camera configuration for a different objective and provide an easy way to define likelihood functions. In real deployment of a camera network for surveillance purposes, covering the total area to a pre-defined level is often not the only requirement. It is also sometimes a necessity that some critical areas such as entrances to prohibited areas are to be monitored by more than one cameras in order to provide redundancy in capturing frontal faces. Shown in Figure 7.4(a) is a modified version of floor plan A (Figure 7.1(a)). The difference between the two is the addition of a number of critical regions which are to be covered by at least two cameras. The total area, as usual, is required to be covered 100%. The likelihood function required in Equation 7.5 can be written as the product of the two likelihoods that correspond to the two separate objectives,

$$\begin{aligned} L\left(\mathbf{r} \mid \phi\right) \quad \propto \quad & \exp\left\{-\left(\mathrm{COV}_{total}\left(\phi\right) - r_{total}\right)^2 / 2\sigma_{total}^2\right\} \\ & \times \exp\left\{-\left(\mathrm{COV}_{crit}\left(\phi\right) - r_{crit}\right)^2 / 2\sigma_{crit}^2\right\}, \end{aligned} \tag{7.12}$$

where COV_{total} and COV_{total} are the percentage of the total area and the percentage of critical regions that are covered by the required number of cameras. r_{total} and r_{crit} are the desired coverage percentage which in this case is 100%. σ_{total} and σ_{crit} is set to 0.1

The parameters used in this experiment were the same as those used in Exp.4(a) which is listed in Table 7.1. The following parameters were used as input to



(a) Floor plan with critical regions

(b) Optimal placement strategy

Figure 7.4: Optimal placement strategy with critical regions covered by at least 2 cameras and other area covered by at least 1 camera.

the proposed TDSA algorithm to compute the optimal camera configurations: $T_0 = 10$, $T_e = 10^{-4}$, $\rho = 0.99$, $\gamma = 10^{-5}$, $n_{max} = 200$, $p_b = p_d = 0.2$.

The resultant placement strategy is shown in Figure 7.4(b) and the total number of cameras used is 27. Since the total number of candidate cameras is 5472, the true optimal strategy was not computable using BIP. However, as can be seen from Figure 7.4(b), the whole area is covered by at least one camera and those critical regions are fully covered by at least 2 cameras.

7.6.4 Cooling Coefficient

Another set of experiments were conducted to examine the performance of the proposed TDSA algorithm for values of the cooling coefficient $\rho \in \{0.5, 0.6, 0.7, 0.8, 0.9, 0.95, 0.99\}$ using both floor plan A and floor plan B. The location of the cameras were sampled every 30 pixels in both x and y directions and their orientations were sampled every 20° degrees. All cameras have FoVs of 120° and depth of 150 pixels. The statistics are summarised in Table 7.2.

It can be seen for Table 7.2 that there is a clear trend: the number of optimal cameras decreases as the coefficient of the cooling schedule increases from 0.5 to 0.95. The better performance in the number of cameras and coverage overlap is due to the nature of simulated annealing based algorithms: A smaller coefficient leads to rapid jumps in temperature which means that if the near-optima are not sampled at a particular temperature, they will be much less likely to be sampled at a lower temperature as the temperature will have decreased so much. Therefore there are higher probability of missing the near-optima for systems with smaller cooling coefficients. On the other hand, a slow schedule (larger cooling coefficient) leads to better near-optima solutions at the cost of longer processing time.

Table 7.2: Effect of varying the cooling coefficient ρ.

Exp.	Cam. loc sep.	ρ	Floor	Num. cand. No.	Num. opt. cam	Cov.
Exp. 5(a)	30	0.5	A	5004	40	96.9%
Exp. 5(b)			B	4518	77	93.6%
Exp. 6(a)	30	0.6	A	5004	37	96.6%
Exp. 6(b)			B	4518	73	94.5%
Exp. 7(a)	30	0.7	A	5004	33	96.3%
Exp. 7(b)			B	4518	63	94.3%
Exp. 8(a)	30	0.8	A	5004	29	96.7%
Exp. 8(b)			B	4518	68	96.7%
Exp. 9(a)	30	0.9	A	5004	21	97.1%
Exp. 9(b)			B	4518	50	96.0%
Exp. 10(a)	30	0.95	A	5004	18	97.8%
Exp. 10(b)			B	4518	34	97.1%
Exp. 11(a)	30	0.99	A	5004	18	98.1%
Exp. 11(b)			B	4518	31	97.0%

7.7 Chapter Summary

In this chapter, an approach to the problem of determining the optimal camera configurations in multi-camera systems has been presented. This is an important and still unsolved problem and optimal solutions will be of significant benefit

in the design of multi-camera surveillance networks. The proposed approach includes a generalised statistical formulation of the problem, taking into account a set of user constraints, the number of cameras and the parameters of the cameras. A trans-dimensional simulated annealing algorithm has been designed to compute the optimal configuration. To evaluate the performance, the proposed approach has been compared with a state-of-the-art method described in [33] and results show that similar performance to the optimal BIP solution can be obtained. Compared to the other two heuristics designed to cope with larger scale problems [56, 128] when BIP cannot handle, the configuration computed by the proposed approach requires 18% less cameras than [56] and 21% less cameras than [128]. In order to demonstrate the flexibility of the proposed approach, a more realistic user objective has been considered: 100% coverage is required for the whole area and a number of critical regions are to be fully covered by at least two cameras. The results show that the proposed approach can successfully solve this problem. Thus, it can be concluded that objective 4 listed in Section 1.2.2 has been successfully achieved.

Chapter 8

Conclusions and Future Work

This thesis has made an attempt to address some of the most common computer vision problems in the context of distributed wireless smart camera networks. In particular, these algorithms tackle the problems of *multi-object tracking and localisation in distributed smart camera networks* and *optimal camera configuration determination*. The DWSC platform has a vast practical application domain but at the same time suffers from tremendous constraints such as limited computational power, bandwidth and energy. The algorithms therefore have been designed in an scalable, flexible, efficient and accurate manner and have taken the dominant constraints of the platform into consideration.

Addressing the first problem of multi-object tracking and localisation requires solving a number of sub-problems, which are calibration of internal parameters, calibration of the camera network in terms of the ground plane homographies for each camera and object handover for tracking. The second problem of camera placement appears naturally when these pervasive devices are put into real use. The locations, orientations, the lens types etc. must be chosen in a way that some kind of user requirement is optimised. More specifically, the research in

these areas has led to four major algorithms:

1. An algorithm which can automatically compute the internal parameters of the camera, without the need for manual input or specific calibration object.

2. An approach which can automatically find the spatial relationship between cameras as well as between each camera and the real-world scene (i.e. the ground plane homography for each camera). The approach is autonomous and accurate for the purpose of object localisation in a large scale distributed wireless smart camera networks.

3. A strategy to deal with the problem of target handover between non-overlapping wireless smart cameras in a multi-object tracking system. The algorithm is scalable and introduces minimal communication overhead.

4. An approach which finds the optimal camera configurations given a user objective (or multiple objectives) and a set of constraints. An example of this task can be the computation of the optimal cameras' positions and orientations to achieve a 100% coverage while minimising the number of cameras used. The approach is able to accommodate a large number of cameras and offers adequate accuracy.

8.1 Self-calibration of the Internal Parameters of Wireless Smart Cameras

Chapter 4 presented an approach for the automatic calibration of low-cost cameras which are assumed to be restricted in their freedom of movement to either pan or tilt movements. Camera parameters, including focal length, principal point, lens distortion and the angle and axis of rotation, can be recovered from

a minimum set of two images captured by the camera, provided that the axis of rotation between the two images goes through the camera's optical centre and is parallel to either the vertical (panning) or horizontal (tilting) axis of the image. In addition, the approach includes a modified RANdom SAmple Consensus (RANSAC) algorithm, as well as improved integration of the radial distortion coefficient in the computation of inter-image homographies. It has been shown that these modifications are able to increase the overall efficiency, reliability and accuracy of the homography computation and hence the calibration procedure using both synthetic and real image sequences.

8.2 Autonomous Calibration of Wireless Smart Camera Networks and Object Localisation

Most current approaches to camera network calibration involves human input, which limits the scalability and flexibility of the networks, prohibiting their use in distributed wireless smart camera networks. To overcome this, Chapter 5 presented a novel approach for calibration of a network of distributed wireless smart cameras covering a large observation area with non-overlapping fields of view, which can then be used for locating and tracking objects.

In the proposed approach, a robot travels through the camera network while updating its position in a global coordinate frame, which it broadcasts to the cameras. The cameras use this, along with the image plane location of the robot, to compute a mapping from their image planes to the global coordinate frame. This is combined with an occupancy map generated by the robot during the mapping process to track the objects. The presented results include a nine node indoor camera network to demonstrate that this autonomous approach is feasible

and offers an acceptable level of accuracy in terms of object locations.

8.3 Object Association for Tracking in Non-overlapping Wireless Smart Cameras

Chapter 6 presented a highly-scalable, distributed strategy to object tracking in wireless smart camera networks with limited resources. In the approach, cameras transmit descriptions of objects to a subset of neighbours, determined using a predictive forwarding strategy. The received descriptions are then matched at the "next" cameras on the objects' path using a probability maximisation process with locally generated descriptions. It has been shown that the predictive forwarding and probabilistic matching strategy can significantly reduce the number of object-misses, ID-switches and ID-losses; it can also vastly reduce the number of required transmissions over the conventional broadcast scenario.

8.4 Statistical Determination of Optimal Camera Configuration

Chapter 7 described a solution to the problem of selecting optimal camera configurations (camera locations, orientations etc.) for multi-camera networks. Previous approaches largely focus on proposing various objective functions to achieve different tasks. Most of them, however, do not generalise well to large scale networks. To tackle this, a statistical formulation of the problem has been introduced and a trans-dimensional simulated annealing algorithm has been proposed to effectively solve the problem. Results show that similar performance to the optimal BIP so-

lution can be obtained on small scale problems and on large scale problems, the proposed approach is more efficient than the two heuristics designed to deal with the problem of BIP.

8.5 Future Work

This dissertation has contributed to several areas of video surveillance in distributed wireless smart cameras networks, however, there are still a number of gaps that can be addressed. Future work that can potential improve the proposed algorithms or open up new avenues of research are summarised below.

- The proposed self-calibration of internal parameters (Chapter 4) may be extended so that the assumption of constant internal parameters can be alleviated. The resultant calibration algorithm can then find its use in more general PTZ cameras.
- The autonomous calibration of camera network (Chapter 5) current only computes the homography between each camera and the ground, which is assumed to be a flat surface. With a more advanced robot the full calibration in terms of cameras' actual position and orientation may be obtainable. When these calibrated cameras are used for object localisation, the assumption that objects move on a common flat surface is no longer needed.
- For the purpose of localisation, a single homography is used (Chapter 5) to locate the foot the object (e.g. people's foot positions). However, it may be feasible to compute a multi-level homography [62, 63] for each camera so that heads of objects are located. For most cameras, head detection is much more accurate than foot detection which is subject to shadows.

- Fusion of multi-camera input has not been a focus of research and may be applied in cases when two cameras share overlapping regions.

- The predicative forward presented in Chapter 6 can be extended to more complex learning and inference algorithms such as probabilistic graphical models.

- The camera placement approach presented in Chapter 7 only considered two different user requirements. Other constraints and requirements can be investigated and incorporated into the generalised formulation and the associated tuning of TDSA parameters to achieve the desirable outcome. The constraints may include maximising the probability of capturing frontal faces, success rate of target hand-off or combinations of different objectives.

- The camera placement approach presented in Chapter 7 can be extended to take 3D environmental and camera models for more realistic results.

Bibliography

[1] A.Criminisi, I.Reid, and A.Zisserman, "A plane measuring device," *Image and Vision Computing*, vol. 17, no. 8, pp. 625–634, 1999.

[2] L. d. Agapito, E. Hayman, and I. D. Reid, "Self-calibration of rotating and zooming cameras," *International Journal of Computer Vision*, vol. 45, no. 2, pp. 107–127, 2001.

[3] H. Akaike, "A new look at the statistical model identification," *Automatic Control, IEEE Transactions on*, vol. 19, no. 6, pp. 716–723, 1974.

[4] C. Andrieu, N. d. Freitas, and A. Doucet, "Reversible jump mcmc simulated annealing for neural networks," in *Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence*, (719898), pp. 11–18, Morgan Kaufmann Publishers Inc., 2000.

[5] N. Anjum and A. Cavallaro, "Localization of distributed wireless cameras," in *Distributed Smart Cameras, 2009. ICDSC 2009. Third ACM/IEEE International Conference on*, pp. 1 –6, 30 2009-sept. 2 2009.

[6] M. Armstrong, A. Zisserman, and P. Beardsley, "Euclidean structure from uncalibrated images," in *Proceedings of the conference on British machine vision (vol. 2)*, (Surrey, UK), pp. 509–518, BMVA Press, 1994.

[7] J. P. Barreto and K. Daniilidis, "Fundamental matrix for cameras with radial distortion," in *Proc. Computer Vision, International Conference on*, vol. 1, pp. 625–632, 2005.

[8] H. Bay, A. Ess, T. Tuytelaars, and L. V. Gool, "Surf: Speeded up robust features," *Computer Vision and Image Understanding (CVIU)*, vol. 110, no. 3, pp. 346–359, 2008.

[9] R. Bodor, A. Drenner, P. Schrater, and N. Papanikolopoulos, "Optimal camera placement for automated surveillance tasks," *Journal of Intelligent & Robotic Systems*, vol. 50, no. 3, pp. 257–295, 2007.

[10] B. Bose and E. Grimson, "Ground plane rectification by tracking moving objects," in *Proceedings of the Joint IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance*, 2003.

[11] M. Bramberger, M. Quaritsch, T. Winkler, B. Rinner, and H. Schwabach, "Integrating multi-camera tracking into a dynamic task allocation system for smart cameras," in *Advanced Video and Signal Based Surveillance, 2005. AVSS 2005. IEEE Conference on*, pp. 474–479, 2005.

[12] S. P. Brooks, N. Friel, and R. King, "Classical model selection via simulated annealing," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 65, no. 2, pp. 503–520, 2003.

[13] M. Brown, R. Hartley, and D. Nister, "Minimal solutions for panoramic stitching," in *Proceedings of the International Conference on Computer Vision and Pattern Recognition (CVPR07)*, pp. 1–8, 2007.

[14] D. Butler, S. Sridharan, and J. Bove, V. M., "Real-time adaptive background segmentation," in *Acoustics, Speech, and Signal Processing, 2003.*

Proceedings. (ICASSP '03). 2003 IEEE International Conference on, vol. 3, pp. III–349–52 vol.3, 2003.

[15] M. Byrod, M. Brown, and K. Astrom, "Minimal solutions for panoramic stitching with radial distortion.," in *BMVC'09*, 2009.

[16] Y. Cai, W. Chen, K. Huang, and T. Tan, "Continuously tracking objects across multiple widely separated cameras," in *Asian Conference on Computer Vision (ACCV 2007)*, pp. 843–852, 2007.

[17] F. Camp, K. Bernardin, and R. Stiefelhagen, "Person tracking in camera networks using graph-based bayesian inference," in *Distributed Smart Cameras, 2009. ICDSC 2009. Third ACM/IEEE International Conference on*, pp. 1–8, 2009.

[18] M. Casares, M. C. Vuran, and S. Velipasalar, "Design of a wireless vision sensor for object tracking in wireless vision sensor networks," in *Distributed Smart Cameras, 2008. ICDSC 2008. Second ACM/IEEE International Conference on*, pp. 1–9, 2008.

[19] Y.-C. Chang, J.-L. Chen, F. Shih, and Y.-T. Wu, "Intelligent wireless sensor network for surveillance systems," in *Consumer Electronics, 2008. ICCE 2008. Digest of Technical Papers. International Conference on*, pp. 1–2, 2008.

[20] P. Chen, P. Ahammad, C. Boyer, I. H. Shih, L. Leon, E. Lobaton, M. Meingast, O. Songhwai, S. Wang, Y. Posu, A. Y. Yang, Y. Chuohao, C. Lung-Chung, J. D. Tygar, and S. S. Sastry, "Citric: A low-bandwidth wireless camera network platform," in *Distributed Smart Cameras, 2008. ICDSC 2008. Second ACM/IEEE International Conference on*, pp. 1–10, 2008.

[21] Y. M. Chi, R. Etienne-Cummings, G. Cauwenberghs, P. Carpenter, and K. Colling, "Video sensor node for low-power ad-hoc wireless networks," in

Information Sciences and Systems, 2007. CISS '07. 41st Annual Conference on, pp. 244–247, 2007.

[22] S.-Y. Chien, S.-Y. Ma, and L.-G. Chen, "Efficient moving object segmentation algorithm using background registration technique," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 12, no. 7, pp. 577–586, 2002.

[23] P. Chulsung and H. C. Pai, "ecam: ultra compact, high data-rate wireless sensor node with a miniature camera," 2006. 1182854 359-360.

[24] O. Chum and J. Matas, "Optimal randomized ransac," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 30, no. 8, pp. 1472–1482, 2008.

[25] N. Cristianini and J. Shawe-Taylor, *An Introduction to Support Vector Machines and other kernel-based learning methods*. Cambridge University, 2000.

[26] D. Culler, D. Estrin, and M. Srivastava, "Guest editors' introduction: Overview of sensor networks," *Computer*, vol. 37, no. 8, pp. 41–49, 2004. 0018-9162.

[27] H.-J. J. E. G. J. R. Del Rincon, J. M. and C. Orrite-Urunuela, "Automatic left luggage detection and tracking using multi-camera ukf.," in *Proceedings of the Internaltional Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance*, pp. 51–58, 2006.

[28] D. Delannay, N. Danhier, and C. De Vleeschouwer, "Detection and recognition of sports(wo)men from multiple views," in *Distributed Smart Cameras, 2009. ICDSC 2009. Third ACM/IEEE International Conference on*, pp. 1 –7, 30 2009-sept. 2 2009.

[29] S. P. Denman, *Improved detection and tracking of objects in surveillance video*. PhD thesis, Queensland University of Technology, 2009.

[30] I. Diaz, M. Heijligers, R. Kleihorst, and A. Danilin, "An embedded low power high efficient object tracker for surveillance systems," in *Distributed Smart Cameras, 2007. ICDSC '07. First ACM/IEEE International Conference on*, pp. 372–378, 2007.

[31] A. Elgammal, D. Harwood, and L. Davis, "Non-parametric model for background subtraction," in *European Conf. Computer Vision*, vol. 2, pp. 751–767, 2000.

[32] T. J. Ellis, D. Makris, and J. K. Black, "Learning a multi-camera topology," in *Joint IEEE Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance (VS-PETS)*, pp. 165–171, 2003.

[33] U. M. Erdem and S. Sclaroff, "Automated camera layout to satisfy task-specific and floor plan-specific coverage requirements," *Computer Vision and Image Understanding*, vol. 103, no. 3, pp. 156–169, 2006.

[34] R. Eshel and Y. Moses, "Homography based multiple camera detection and tracking of people in a dense crowd," in *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pp. 1 –8, june 2008.

[35] O. D. Faugeras, Q. T. Luong, and S. J. Maybank, "Camera self-calibration: Theory and experiments," in *ECCV '92: Proceedings of the Second European Conference on Computer Vision*, pp. 321–334, Springer-Verlag, 1992.

[36] M. A. Fischler and R. C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Comm. of the ACM*, vol. 24, pp. 381–395, 1981.

[37] A. Fitzgibbon, "Simultaneous linear estimation of multiple view geometry and lens distortion," in *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 1, pp. I–125 – I–132, 2001.

[38] S. Fleishman, D. Cohen-Or, and D. Lischinski, "Automatic camera placement for image-based modeling," in *Computer Graphics and Applications, 1999. Proceedings. Seventh Pacific Conference on*, pp. 12–20, 315, fleishman99.

[39] D. Focken and R. Stiefelhagen, "Towards vision-based 3-d people tracking in a smart room," in *Multimodal Interfaces, 2002. Proceedings. Fourth IEEE International Conference on*, pp. 400 – 405, 2002.

[40] L. M. Fuentes and S. A. Velastin, "Tracking people for automatic surveillance applications," *Pattern recognition and image analysis*, vol. 2652/2003, pp. 238–245, 2003.

[41] S. Funiak, C. Guestrin, M. Paskin, and R. Sukthankar, "Distributed localization of networked cameras," in *Information Processing in Sensor Networks, 2006. IPSN 2006. The Fifth International Conference on*, pp. 34 –42, 0-0 2006.

[42] A. Gilbert and R. Bowden, "Tracking objects across cameras by incrementally learning inter-camera colour calibration and patterns of activity," in *2006 European Conference on Computer Vision*, (Gratz, Austria), pp. 125–136, 2006.

[43] O. Gnawali, R. Fonseca, K. Jamieson, D. Moss, and P. Levis, "Collection tree protocol," in *Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems*, SenSys 2009, (New York, NY, USA), pp. 1–14, ACM, 2009.

[44] G. H. Golub and C. F. Van Loan, *Matrix Computations*. The John Hopkins University Press, 1996.

[45] V. Granville, M. Krivanek, and J.-P. Rasson, "Simulated annealing: A proof

of convergence," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 16, no. 6, pp. 652–656, 1994.

[46] P. J. GREEN, "Reversible jump markov chain monte carlo computation and bayesian model determination," *Biometrika*, vol. 82, no. 4, pp. 711–732, 1995.

[47] G. Grisetti, C. Stachniss, and W. Burgard, "Improved techniques for grid mapping with rao-blackwellized particle filters," *Robotics, IEEE Transactions on*, vol. 23, no. 1, pp. 34–46, 2007.

[48] I. Haritaoglu, D. Harwood, and L. S. Davis, "W4: real-time surveillance of people and their activities," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 22, no. 8, pp. 809–830, 2000.

[49] R. I. Hartley, "Self-calibration of stationary cameras," *Int. J. Comput. Vision*, vol. 22, no. 1, pp. 5–23, 1997. 250491.

[50] R. Hartley and P. Sturm, "Triangulation," *Computer Vision and*, vol. 68, pp. 146–157, 1997.

[51] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*. Cambridge University Press New York, NY, USA, 2003.

[52] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. Cambridge University Press, second ed., 2004.

[53] S. Hengstler, D. Prashanth, F. Sufen, and H. Aghajan, "Mesheye: A hybrid-resolution smart camera mote for applications in distributed intelligent surveillance," in *Information Processing in Sensor Networks, 2007. IPSN 2007. 6th International Symposium on*, pp. 360–369, 2007.

[54] A. Heyden and K. Astrom, "Euclidean reconstruction from constant intrinsic

parameters," in *Pattern Recognition, 1996., Proceedings of the 13th International Conference on*, vol. 1, pp. 339–343, aug 1996.

[55] M. Hoffmann, M. Wittke, Y. Bernard, R. Soleymani, and J. Hahner, "Dmctrac: Distributed multi camera tracking," in *Distributed Smart Cameras, 2008. ICDSC 2008. Second ACM/IEEE International Conference on*, pp. 1–10, 2008.

[56] E. Horster and R. Lienhart, *Multi-Camera Networks: Concepts and Applications*, ch. 5, Optimal Placement of Multiple Visual Sensors. Elsevier, 2009.

[57] T. Huang and S. Russell, "Object identification in a bayesian context," in *In Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence (IJCAI-97*, pp. 1276–1283, Morgan Kaufmann, 1997.

[58] O. Javed, Z. Rasheed, K. Shafique, and M. Shah, "Tracking across multiple cameras with disjoint views," in *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, pp. 952–957 vol.2, 2003.

[59] O. Javed, K. Shafique, Z. Rasheed, and M. Shah, "Modeling inter-camera spacectime and appearance relationships for tracking across non-overlapping views," *Computer Vision and Image Understanding*, vol. 109, no. 2, pp. 146 – 162, 2008.

[60] S.-W. Joo and Q. Zheng, "A temporal variance-based moving target detector," in *IEEE International Workshop on Performance Evaluation of Tracking and Surveillance (PETS)*, 2005.

[61] J. Kang, I. Cohen, and G. Medioni, "Multi-views tracking within and across uncalibrated camera streams," in *First ACM SIGMM international workshop on Video surveillance*, IWVS '03, (New York, NY, USA), pp. 21–33, ACM, 2003.

[62] S. Khan and M. Shah, "A multiview approach to tracking people in crowded scenes using a planar homography constraint," in *European Conference on Computer Vision* (A. Leonardis, H. Bischof, and A. Pinz, eds.), vol. 3954, pp. 133–146, Springer Berlin / Heidelberg, 2006.

[63] S. Khan and M. Shah, "Tracking multiple occluding people by localizing on multiple scene planes," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 31, pp. 505 –519, march 2009.

[64] K. Kim, T. H. Chalidabhongse, D. Harwood, and L. Davis, "Background modeling and subtraction by codebook construction," in *Image Processing, 2004. ICIP '04. 2004 International Conference on*, vol. 5, pp. 3061–3064 Vol. 5, 2004.

[65] K. Kim and L. S. Davis, "Multi-camera tracking and segmentation of occluded people on ground plane using search-guided particle filtering," in *European Conference on Computer Vision*, pp. 98–109, 2006.

[66] H. Kim, J. Romberg, and W. Wolf, "Multi-camera tracking on a graph using markov chain monte carlo," in *Distributed Smart Cameras, 2009. ICDSC 2009. Third ACM/IEEE International Conference on*, pp. 1–8, 2009.

[67] S. Kirkpatrick, "Optimization by simulated annealing: Quantitative studies," *Journal of Statistical Physics*, vol. 34, no. 5, pp. 975–986, 1984.

[68] T. H. Ko and N. M. Berry, "Distributed calibration and tracking with low-power image sensors," in *Diversity in Computing Conference, 2005 Richard Tapia Celebration of*, pp. 40–43, 2005.

[69] T. Ko, Z. M. Charbiwala, S. Ahmadian, M. Rahimi, M. B. Srivastava, S. Soatto, and D. Estrin, "Exploring tradeoffs in accuracy, energy and latency of scale invariant feature transform in wireless camera networks," in

Distributed Smart Cameras, 2007. ICDSC '07. First ACM/IEEE International Conference on, pp. 313–320, 2007.

[70] B. Krishnamachari, *Networking Wireless Sensors*. Cambridge University Press, 2005.

[71] R. Lakemond, C. Fookes, and S. Sridharan, "Practical improvements to simultaneous computation of multi-view geometry and radial lens distortion," 2011.

[72] X. Liu, P. Kulkarni, P. Shenoy, and D. Ganesan, "Snapshot: A self-calibration protocol for camera sensor networks," in *Broadband Communications, Networks and Systems, 2006. BROADNETS 2006. 3rd International Conference on*, pp. 1–10, 2006.

[73] L. Lo Presti and M. La Cascia, "Real-time object detection in embedded video surveillance systems," in *Image Analysis for Multimedia Interactive Services, 2008. WIAMIS '08. Ninth International Workshop on*, pp. 151–154, 2008.

[74] Q. T. Luong and O. D. Faugeras, "Self-calibration of a moving camera from point correspondences and fundamental matrices," *Int. J. Comput. Vision*, vol. 22, no. 3, pp. 261–289, 1997.

[75] C. Madden, E. D. Cheng, and M. Piccardi, "Tracking people across disjoint camera views by an illumination-tolerant appearance representation," *Mach. Vision Appl.*, vol. 18, no. 3, pp. 233–247, 2007.

[76] P. Marquez-Neila, J. Garcia Miro, J. M. Buenaposada, and L. Baumela, "Improving ransac for fast landmark recognition," in *Computer Vision and Pattern Recognition Workshops, 2008. CVPRW '08. IEEE Computer Society Conference on*, pp. 1–8, 2008.

[77] J. Matas and O. Chum, "Randomized ransac with td,d test," *Image and Vision Computing*, vol. 22, no. 10, pp. 837–842, 2004.

[78] J. Matas and O. Chum, "Randomized ransac with sequential probability ratio test," in *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, vol. 2, pp. 1727–1732, 2005.

[79] H. Medeiros, J. Park, and A. Kak, "Distributed object tracking using a cluster-based kalman filter in wireless camera networks," *Selected Topics in Signal Processing, IEEE Journal of*, vol. 2, no. 4, pp. 448–463, 2008.

[80] A. Mittal and L. S. Davis, "M2tracker: A multi-view approach to segmenting and tracking people in a cluttered scene using region-based stereo," in *European Conference on Computer Vision*, pp. 189–203, 2002.

[81] A. Mittal and L. Davis, *Visibility Analysis and Sensor Planning in Dynamic Environments Computer Vision - ECCV 2004*, vol. 3021 of *Lecture Notes in Computer Science*, pp. 175–189. Springer Berlin / Heidelberg, 2004.

[82] A. Mittal and L. S. Davis, "A general method for sensor planning in multi-sensor systems: Extension to random occlusion," *Int. J. Comput. Vision*, vol. 76, no. 1, pp. 31–52, 2008.

[83] B. Mičušík and T. Pajdla, "Estimation of omnidirectional camera model from epipolar geometry," in *Proc. Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, vol. 1, pp. 485–490, 2003.

[84] R. Mohammad, B. Rick, I. I. Obimdinachi, C. G. Juan, W. Jay, E. Deborah, and S. Mani, "Cyclops: in situ image sensing and interpretation in wireless sensor networks," 2005. 1098939 192-204.

[85] P. Mordohai and G. Medioni, "Dense multiple view stereo with general camera placement using tensor voting," in *3D Data Processing, Visualization and*

Transmission, 2004. 3DPVT 2004. Proceedings. 2nd International Symposium on, pp. 725–732, 2004.

[86] D. Nister, "Preemptive ransac for live structure and motion estimation," *Mach. Vision Appl.*, vol. 16, no. 5, pp. 321–329, 2005.

[87] G. Olague and R. Mohr, "Optimal camera placement to obtain accurate 3d point positions," in *Pattern Recognition, 1998. Proceedings. Fourteenth International Conference on*, vol. 1, pp. 8–10 vol.1, 1998.

[88] J. O'Rourke, *Art gallery theorems and algorithms*. Oxford University Press, Inc., 1987.

[89] D. O'Rourke, R. Jurdak, J. Liu, D. Moore, and T. Wark, "On the feasibility of using servo-mechanisms in wireless multimedia sensor network deployments," in *The 4th IEEE International Workshop on Practical Issues In Building Sensor Network Applications (SenseApp 2009) held in conjunction with LCN*, (Zurich, Switzerland), pp. 826–833, IEEE Press, 2009.

[90] K. A. Patwardhan, G. Sapiro, and V. Morellas, "Robust foreground detection in video using pixel layers," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 30, no. 4, pp. 746–751, 2008.

[91] R. Pflugfelder and H. Bischof, "Localization and trajectory reconstruction in surveillance cameras with nonoverlapping views," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 32, pp. 709 –721, april 2010.

[92] M. Pollefeys and L. Van Gool, "A stratified approach to metric self-calibration," in *Computer Vision and Pattern Recognition, 1997. Proceedings., 1997 IEEE Computer Society Conference on*, pp. 407–412, 1997.

[93] M. Pollefeys, L. Van Gool, and A. Oosterlinck, "The modulus constraint: a new constraint self-calibration," in *Pattern Recognition, 1996., Proceedings of the 13th International Conference on*, vol. 1, pp. 349–353, aug 1996.

[94] K. Purushottam, G. Deepak, S. Prashant, and L. Qifeng, "Senseye: a multi-tier camera sensor network," 2005. 1101191 229-238.

[95] M. Quaritsch, M. Kreuzthaler, B. Rinner, H. Bischof, and B. Strobl, "Autonomous multicamera tracking on embedded smart cameras," *EURASIP J. Embedded Syst.*, vol. 2007, pp. 35–35, Jan. 2007.

[96] A. Rahimi, B. Dunagan, and T. Darrell, "Simultaneous calibration and tracking with a network of non-overlapping sensors," in *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, vol. 1, pp. I–187–I–194 Vol.1, 2004.

[97] S. Ram, K. R. Ramakrishnan, P. K. Atrey, V. K. Singh, and M. S. Kankanhalli, "A design methodology for selection and placement of sensors in multimedia surveillance systems," in *Proceedings of the 4th ACM international workshop on Video surveillance and sensor networks*, (1178801), pp. 121–130, ACM, 2006.

[98] I. Rekleitis, D. Meger, and G. Dudek, "Simultaneous planning, localization, and mapping in a camera sensor network," *Robotics and Autonomous Systems*, vol. 54, no. 11, pp. 921–932, 2006.

[99] B. Rinner and W. Wolf, "An introduction to distributed smart cameras," *Proceedings of the IEEE*, vol. 96, pp. 1565 –1575, oct. 2008.

[100] J. Rissanen, "Stochastic complexity," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 49, no. 3, pp. 223–239, 1987.

[101] K. Romer and F. Mattern, "The design space of wireless sensor networks," *Wireless Communications, IEEE*, vol. 11, no. 6, pp. 54–61, 2004. 1536-1284.

[102] A. Rowe, D. Goel, and R. Rajkumar, "Firefly mosaic: A vision-enabled wireless sensor networking system," in *Real-Time Systems Symposium, 2007. RTSS 2007. 28th IEEE International*, pp. 459–468, 2007.

[103] A. C. Sankaranarayanan and R. Chellappa, "Optimal multi-view fusion of object locations," in *Motion and video Computing, 2008. WMVC 2008. IEEE Workshop on*, pp. 1–8, 2008.

[104] G. Schwarz, "Estimating the dimension of a model," *The Annals of Statistics*, vol. 6, no. 2, pp. 461–464, 1985.

[105] Y. Shen, W. Hu, J. Liu, M. Yang, and C. T. Chou, "Efficient background subtraction for real-time tracking in embedded camera networks," in *Submitted to: The 10th ACM Conference on Embedded Networked Sensor Systems (SenSys 2012)*, 2012.

[106] S. Sridharan, C. Fookes, A. Goonetilleke, G. Mamic, C. McCool, F. Lin, J. Cook, D. Mckinnon, S. Denman, R. Lakemond, D. Chen, B. Chen, P.-c. Chou, C. Davoren, A. Yau, P. Lucey, D. Dean, and D. Ryan, "Intelligent surveillance system and early warning support for monitoring human traffic," tech. rep., Queensland University of Technology, 2008.

[107] C. Stauffer and W. E. L. Grimson, "Adaptive background mixture models for real-time tracking," in *Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on.*, vol. 2, p. 252 Vol. 2, 1999.

[108] C. Stauffer and T. Kinh, "Automated multi-camera planar tracking correspondence modeling," in *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on*, vol. 1, pp. 259–266, june 2003.

[109] M. R. Steele and C. Jaynes, "Overconstrained linear estimation of radial distortion and multi-view geometry," in *Computer Vision ECCV 2006*,

vol. 3951/2006 of *Lecture Notes in Computer Science*, pp. 253–264, Berlin / Heidelberg: Springer, 2006.

[110] P. Sturm, "Critical motion sequences for monocular self-calibration and uncalibrated euclidean reconstruction," in *Computer Vision and Pattern Recognition, 1997. Proceedings., 1997 IEEE Computer Society Conference on*, pp. 1100 –1105, jun 1997.

[111] P. Sturm, Z. L. Cheng, P. C. Y. Chen, and A. N. Poo, "Focal length calibration from two views: method and analysis of singular cases," *Computer Vision and Image Understanding*, vol. 99, no. 1, pp. 58–95, 2005.

[112] M. Taj and A. Cavallaro, "Multi-camera track-before-detect," in *Distributed Smart Cameras, 2009. ICDSC 2009. Third ACM/IEEE International Conference on*, pp. 1–6, 30 2009-sept. 2 2009.

[113] S. Thrun, W. Burgard, and D. Fox, *Probabilistic robotics*. MIT Press, 2005.

[114] K. Tieu, G. Dalley, and W. E. L. Grimson, "Inference of non-overlapping camera network topology by measuring statistical dependence," in *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, vol. 2, pp. 1842–1849 Vol. 2, 2005.

[115] B. Tordoff and D. W. Murray, "Violating rotating camera geometry: The effect of radial distortion on self-calibration," in *15th International Conference on Pattern Recognition (ICPR'00)* (W. M. David, ed.), vol. 1, (Barcelona, Spain), pp. 1423–1423, 2000.

[116] B. Tordoff and D. W. Murray, "The impact of radial distortion on the self-calibration of rotating cameras," *Computer Vision and Image Understanding*, vol. 96, no. 1, pp. 17–34, 2004.

[117] B. J. Tordoff and D. W. Murray, "Guided-mlesac: faster image transform estimation by using matching priors," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 27, no. 10, pp. 1523–1535, 2005.

[118] P. H. S. Torr and A. Zisserman, "Mlesac: a new robust estimator with application to estimating image geometry," *Computer Vision and Image Understanding*, vol. 78, no. 1, pp. 138–156, 2000.

[119] B. Triggs, "Autocalibration from planar scenes," in *Computer Vision - ECCV'98* (H. Burkhardt and B. Neumann, eds.), vol. 1406 of *Lecture Notes in Computer Science*, pp. 89–105, Springer Berlin / Heidelberg, 1998.

[120] T. Tuytelaars and K. Mikolajczyk, "Local invariant feature detectors: A survey," *Foundations and Trends in Computer Graphics and Vision*, vol. 3, no. 3, pp. 177–280, 2007.

[121] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, vol. 1, pp. I–511–I–518 vol.1, 2001.

[122] X. Wang, S. Wang, and J. Ma, "Multi-view tracking in wireless sensor networks," in *Intelligent Control and Automation, 2006. WCICA 2006. The Sixth World Congress on*, vol. 1, pp. 5115–5119, 2006.

[123] T. Wark, P. Corke, J. Liu, and D. Moore, "Design and evaluation of an image analysis platform for low-power, low-bandwidth camera networks," in *Workshop on Applications, Systems, and Algorithms for Image Sensing 2008*, 2008.

[124] Y. Yao, C.-H. Chen, B. Abidi, D. Page, A. Koschan, and M. Abidi, "Can you see me now? sensor positioning for automated and persistent surveil-

lance," *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, vol. 40, no. 1, pp. 101–115, 2010.

[125] J. Yao and J.-M. Odobez, *Multi-Camera Networks: Concepts and Applications*, ch. 15, Multi-Person Bayesian Tracking with Multiple Cameras, pp. 363–388. Elsevier, 2009.

[126] A. Yilmaz, O. Javed, and M. Shah, "Object tracking: A survey," *ACM Comput. Surv.*, vol. 38, Dec. 2006.

[127] Z. Zhang and T. Kanade, "Determining the epipolar geometry and its uncertainty: A review," *International Journal of Computer Vision*, vol. 27, pp. 161–195, 1998.

[128] J. Zhao, S.-c. S. Cheung, and T. Nguyen, *Multi-Camera Networks: Concepts and Applications*, ch. 6, Optimal Visual Sensor Network Configuration. Elsevier, 2009.

[129] T. Zhao and R. Nevatia, "Tracking multiple humans in complex situations," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 26, no. 9, pp. 1208–1221, 2004.

[130] Z. Zhaoxiang, M. Li, K. Huang, and T. Tan, "Robust automated ground plane rectification based on moving vehicles for traffic scene surveillance," in *Image Processing, 2008. ICIP 2008. 15th IEEE International Conference on*, pp. 1364–1367, 2008.

[131] A. Zisserman, D. Liebowitz, and M. Armstrong, "Resolving ambiguities in auto-calibration," *Philosophical Transactions of the Royal Society London*, vol. A, no. 356(1740), pp. 1193–1211, 1998.